INF10004 Database Analysis and Design: Assignment 01

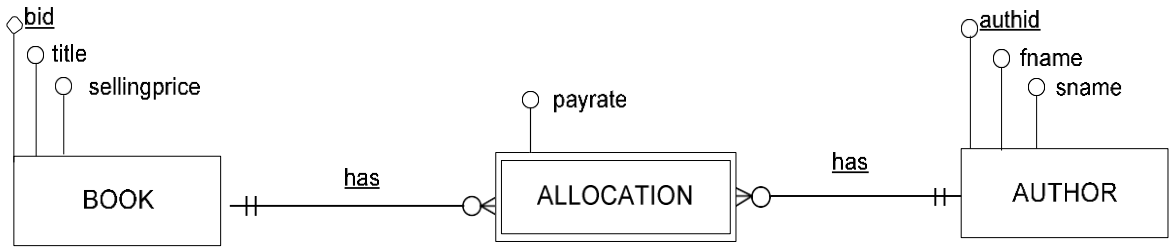
This is a group assignment with a maximum of 3-4 students per group

Assessment Value = 10% (see blackboard for due date):

Submission Requirements

- Create a text file that contains all the statements required for all tasks of this assignment.
- Your scripts must work with Oracle iSQL Junior
- You must submit your assignment via BB by the due date. (Note: Late submissions **will** attract a penalty). Only one submission per group allowed.
- Group must **demonstrate** the assignment to your tutor using your script using ISQL Junior. Make sure that you have a soft copy of the script with you in the lab.

A database analyst has developed the following ER Diagram.



Task 1

1A. Create Text File

Create a text file named ASSIGNMENT_01.TXT (Use notepad++ to create a TXT file and then choose the language as SQL)

Add group member details: Student ID(s) and Student name(s) the first line of this file.

e.g. /* 17385606 Chandrika Kumaratunga

12435689 Mahinda Rajapaksa */

1B. Create Tables and Constraints

Write **Create Table SQL statements** based on the above ERD.

- All tables must have primary keys.
- All tables must have appropriate foreign key constraints.
- Foreign keys must have the same name, datatype and size of the primary key that they refer to
- The following columns datatypes and sizes must be used

bid, authid	number(4)
title, sname, fname	varchar2(30)
sellingprice, payrate	number(6,2)

- The following constraints must be applied to the appropriate tables. All unique and check constraints must be named.

Table	Column	Type	Range
author	sname & fname	Unique	
book	title	Not Null	
book	sellingprice	Check	not negative
allocation	payrate	Check	1 to 79.99

Append the SQL statements to the ASS2.TXT file.

Prefix each CREATE TABLE statement with a **prompt** statement.

e.g.

```
Prompt Creating Table Employee;
CREATE TABLE
  Employee(...
  empid NUMBER(3),
  empname  VARCHAR2(20) Not Null,
  empage  NUMBER(3),
  Primary Key (empid),
  CONSTRAINT UC_EMPAGE UNIQUE (empage),
...
);
```

1C. Valid Insert Statements

Append **SQL Insert statements** to the .TXT file that will add the data to the **3 tables** based on the following information:

This sample data has been obtained from **handwritten documents** supplied by the client. The data below has not been stored or obtained from a computer system.

Table 1 Author Details

Author ID	Surname	First Name
40	Ziggle	Carl
42	Taylor	Tayla
44	Merdovic	Damir
45	Grossman	Paul
47	Ziggle	Annie
48	Zhao	Cheng
50	Phan	Annie

Table 2 Book Details

Book ID	Book Title	Selling Price
101	Knitting with Dog Hair	6.99
105	Avoiding Large Ships	11
107	Dealing with stuff	6.5
108	Teach fish to sing	10.99
109	Guide to hands free texting	10.5
113	You call that a lecture?	17.5

Table 3 Allocation Details

Item ID	Author	Pay Rate
101	42	$25
101	45	$32
108	47	$35
113	48	$40
109	47	$42
105	42	$26
105	47	$25
105	40	$19
107	42	$35
108	40	$45

1D. Invalid insert statements

Write 4 Insert that will fail because of data constraints.

Statement 1 must fail due when adding a **duplicate combination of author sname & fname**.

Statement 2 must fail due when adding a textboox with a **null title.**

Statement 3 must fail due when adding a book with a **negative selling price.**

Statement 4 must fail due when adding an allocation **out of range payrate.**.

Append **SQL Insert statements** to the .TXT file.

Prefix each statement with **comment (use - - symbol)**.

1E. SQL Queries

Write the following queries and add each to the .TXT file. Prefix each statement with an appropriate **comment (use - - symbol)** so the SQL statement can correctly identify.

Query	Requirement	Sample Output
1	Write a query that lists rows from the allocation table. Columns: • book id • author id • payment rate Sequence: • book id • author id	101 42 25 101 45 32 105 40 19 105 42 26 105 47 25 107 42 35 108 40 45 108 47 35 109 47 42 113 48 40
2	Write a query that lists rows from the allocation and book tables. Columns: • book id • book title • author id • payment rate Sequence: • book id • author id	101 Knitting with Dog Hair 42 25 101 Knitting with Dog Hair 45 32 105 Avoiding Large Ships 40 19 105 Avoiding Large Ships 42 26 105 Avoiding Large Ships 47 25 107 Dealing with stuff 42 35 108 Teach fish to sing 40 45 108 Teach fish to sing 47 35 109 Guide to hands free texting 47 42 113 You call that a lecture? 48 40

<table>
<tr><td>3</td><td>Write a query that lists rows from the allocation, book and author tables.
Columns:
• book id
• book title
• book price
• author id
• author name
• payment rate
Sequence:
• book id
• author id</td><td>
<table>
<tr><td>101</td><td>Knitting with Dog Hair</td><td>6.99</td><td>42</td><td>Taylor</td></tr>
<tr><td>101</td><td>Knitting with Dog Hair</td><td>6.99</td><td>45</td><td>Grossman</td></tr>
<tr><td>105</td><td>Avoiding Large Ships</td><td>11</td><td>40</td><td>Ziggle</td></tr>
<tr><td>105</td><td>Avoiding Large Ships</td><td>11</td><td>42</td><td>Taylor</td></tr>
<tr><td>105</td><td>Avoiding Large Ships</td><td>11</td><td>47</td><td>Ziggle</td></tr>
<tr><td>107</td><td>Dealing with stuff</td><td>6.5</td><td>42</td><td>Taylor</td></tr>
<tr><td>108</td><td>Teach fish to sing</td><td>10.99</td><td>40</td><td>Ziggle</td></tr>
<tr><td>108</td><td>Teach fish to sing</td><td>10.99</td><td>47</td><td>Ziggle</td></tr>
<tr><td>109</td><td>Guide to hands free texting</td><td>10.5</td><td>47</td><td>Ziggle</td></tr>
<tr><td>113</td><td>You call that a lecture?</td><td>17.5</td><td>48</td><td>Zhao</td></tr>
</table>
</td></tr>
<tr><td>4</td><td>Write a query that lists the average price of rows in the book table.

The heading must match the example shown.</td><td>
<table>
<tr><th>Average Price</th></tr>
<tr><td>10.58</td></tr>
</table>
</td></tr>
<tr><td>5</td><td>Write a query that lists rows from the book table
Columns:
• book id
• book title
• book price
Restriction:
• Only list rows that have a price less than the average price
Sequence:
• book id</td><td>
<table>
<tr><td>101</td><td>Knitting with Dog Hair</td><td>6.99</td></tr>
<tr><td>107</td><td>Dealing with stuff</td><td>6.5</td></tr>
<tr><td>109</td><td>Guide to hands free texting</td><td>10.5</td></tr>
</table>
</td></tr>
<tr><td>6</td><td>Write a query that counts the total number of rows in the allocation table grouped by book id Columns:
• book id
• count
Sequence:
• count
• book id</td><td>
<table>
<tr><th>BID</th><th>COUNT(*)</th></tr>
<tr><td>107</td><td>1</td></tr>
<tr><td>109</td><td>1</td></tr>
<tr><td>113</td><td>1</td></tr>
<tr><td>101</td><td>2</td></tr>
<tr><td>108</td><td>2</td></tr>
<tr><td>105</td><td>3</td></tr>
</table>
</td></tr>
<tr><td>7</td><td>Write a query that counts the total number of rows in the allocation table grouped by book id Columns:
• book id
• book title
• count
Sequence:
• count
• book id</td><td>
<table>
<tr><th>BID</th><th>TITLE</th><th>COUNT(*)</th></tr>
<tr><td>107</td><td>Dealing with stuff</td><td>1</td></tr>
<tr><td>109</td><td>Guide to hands free texting</td><td>1</td></tr>
<tr><td>113</td><td>You call that a lecture?</td><td>1</td></tr>
<tr><td>101</td><td>Knitting with Dog Hair</td><td>2</td></tr>
<tr><td>108</td><td>Teach fish to sing</td><td>2</td></tr>
<tr><td>105</td><td>Avoiding Large Ships</td><td>3</td></tr>
</table>
</td></tr>
</table>

<table>
<tr>
<td>8</td>
<td>Write a query that counts the total number of rows in the allocation table grouped by book id.
Only show totals greater than 1
Columns:
• book id
• book title
• count
Sequence:
• count
• book id</td>
<td>
<table>
<tr><th>BID</th><th>TITLE</th><th>COUNT(*)</th></tr>
<tr><td>101</td><td>Knitting with Dog Hair</td><td>2</td></tr>
<tr><td>108</td><td>Teach fish to sing</td><td>2</td></tr>
<tr><td>105</td><td>Avoiding Large Ships</td><td>3</td></tr>
</table>
</td>
</tr>
<tr>
<td>9</td>
<td>Write a query that lists rows from the allocation and author tables.
Columns:
• author id
• author name
• book id
Sequence:
• author id
• book id</td>
<td>
<table>
<tr><th>AUTHID</th><th>SNAME</th><th>FNAME</th><th>BID</th></tr>
<tr><td>40</td><td>Ziggle</td><td>Carl</td><td>105</td></tr>
<tr><td>40</td><td>Ziggle</td><td>Carl</td><td>108</td></tr>
<tr><td>42</td><td>Taylor</td><td>Tayla</td><td>101</td></tr>
<tr><td>42</td><td>Taylor</td><td>Tayla</td><td>105</td></tr>
<tr><td>42</td><td>Taylor</td><td>Tayla</td><td>107</td></tr>
<tr><td>45</td><td>Grossman</td><td>Paul</td><td>101</td></tr>
<tr><td>47</td><td>Ziggle</td><td>Annie</td><td>105</td></tr>
<tr><td>47</td><td>Ziggle</td><td>Annie</td><td>108</td></tr>
<tr><td>47</td><td>Ziggle</td><td>Annie</td><td>109</td></tr>
<tr><td>48</td><td>Zhao</td><td>Cheng</td><td>113</td></tr>
</table>
</td>
</tr>
<tr>
<td>10</td>
<td>Write a query that lists rows from the allocation and author tables.
Columns:
• author id
• author name
• book id
Sequence:
• author id
• book id

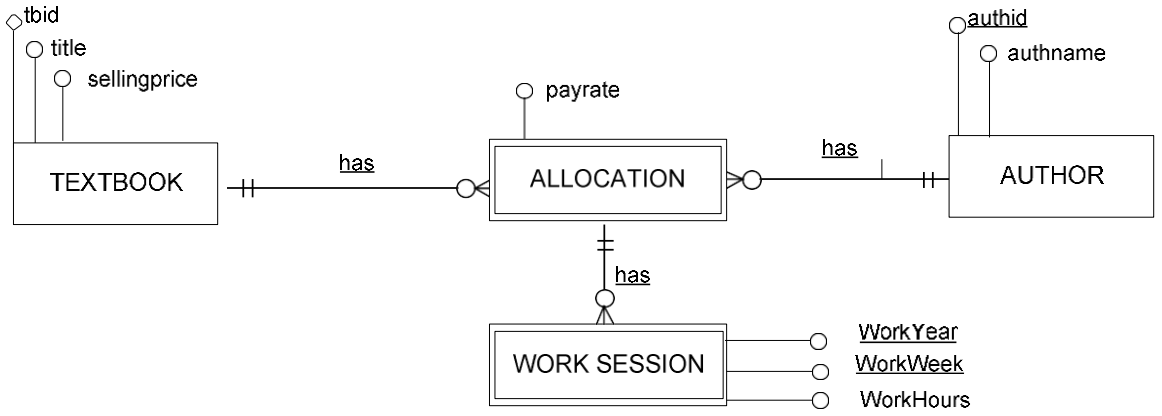
Ensure that all authors are listed even if do not included in the allocation table</td>
<td>
<table>
<tr><th>AUTHID</th><th>SNAME</th><th>FNAME</th><th>BID</th></tr>
<tr><td>40</td><td>Ziggle</td><td>Carl</td><td>105</td></tr>
<tr><td>40</td><td>Ziggle</td><td>Carl</td><td>108</td></tr>
<tr><td>42</td><td>Taylor</td><td>Tayla</td><td>101</td></tr>
<tr><td>42</td><td>Taylor</td><td>Tayla</td><td>105</td></tr>
<tr><td>42</td><td>Taylor</td><td>Tayla</td><td>107</td></tr>
<tr><td>44</td><td>Merdovic</td><td>Damir</td><td></td></tr>
<tr><td>45</td><td>Grossman</td><td>Paul</td><td>101</td></tr>
<tr><td>47</td><td>Ziggle</td><td>Annie</td><td>105</td></tr>
<tr><td>47</td><td>Ziggle</td><td>Annie</td><td>108</td></tr>
<tr><td>47</td><td>Ziggle</td><td>Annie</td><td>109</td></tr>
<tr><td>48</td><td>Zhao</td><td>Cheng</td><td>113</td></tr>
<tr><td>50</td><td>Phan</td><td>Annie</td><td></td></tr>
</table>
</td>
</tr>
<tr>
<td>11</td>
<td>Write a query that lists rows from the allocation, book and author tables.
Columns:
• author id
• author name
• book id
• book title
Sequence:
• author id
• book id

Ensure that all authors are listed
- even if they are not included in the allocation table</td>
<td>
<table>
<tr><th>AUTHID</th><th>SNAME</th><th>FNAME</th><th>BID</th><th>TITLE</th></tr>
<tr><td>40</td><td>Ziggle</td><td>Carl</td><td>105</td><td>Avoiding Large Ships</td></tr>
<tr><td>40</td><td>Ziggle</td><td>Carl</td><td>108</td><td>Teach fish to sing</td></tr>
<tr><td>42</td><td>Taylor</td><td>Tayla</td><td>101</td><td>Knitting with Dog Hair</td></tr>
<tr><td>42</td><td>Taylor</td><td>Tayla</td><td>105</td><td>Avoiding Large Ships</td></tr>
<tr><td>42</td><td>Taylor</td><td>Tayla</td><td>107</td><td>Dealing with stuff</td></tr>
<tr><td>44</td><td>Merdovic</td><td>Damir</td><td></td><td></td></tr>
<tr><td>45</td><td>Grossman</td><td>Paul</td><td>101</td><td>Knitting with Dog Hair</td></tr>
<tr><td>47</td><td>Ziggle</td><td>Annie</td><td>105</td><td>Avoiding Large Ships</td></tr>
<tr><td>47</td><td>Ziggle</td><td>Annie</td><td>108</td><td>Teach fish to sing</td></tr>
<tr><td>47</td><td>Ziggle</td><td>Annie</td><td>109</td><td>Guide to hands free texting</td></tr>
<tr><td>48</td><td>Zhao</td><td>Cheng</td><td>113</td><td>You call that a lecture?</td></tr>
<tr><td>50</td><td>Phan</td><td>Annie</td><td></td><td></td></tr>
</table>
</td>
</tr>
</table>

Task 2:

The ERD has been modified. A new entity named WorkSession has been added.

This entity is used to record the number of hours worked in a week by an author on the development of a book



2A. Create Tables and Constraints

Write **Create Table SQL statements** Work Session table that you have created

- The table must have a primary key.
- The table must have appropriate foreign key constraints.
- Foreign keys must have the same name, datatype and size of the primary key that they refer to
- The following columns datatypes and sizes and named constraints must be used

Column	DataType	Constraint
WorkYear	number(4)	Range 2011 - 2013
WorkWeek	number(2)	Range 1-52
WorkHours	number(4,2)	Range 0.5 – 99.99

Append the SQL statements to the .TXT file.

Prefix the CREATE TABLE statement with a **comment**.

2B. Valid Insert Statements

Append **SQL Insert statements** to the .TXT file that will add the data to the **worksession table** based on the following information:

Item Id	Author Id	Year	Week	Hours
101	42	2012	5	5
101	42	2012	6	4
101	42	2012	7	5
101	45	2012	5	10
101	45	2012	7	10
105	42	2012	5	6
105	47	2012	4	8
105	47	2012	6	7
105	47	2012	8	8
108	40	2011	52	4
108	40	2012	4	15
108	40	2012	6	6
108	47	2012	8	4
109	47	2012	5	5
109	47	2012	6	5
113	48	2012	10	15
113	48	2012	11	4
113	48	2012	12	1

2C. Invalid Insert Statements

Write five **Insert** SQL statements that **must fail** because of foreign key constraints and data constraints. Append **SQL Insert statements** to the .TXT file.

Prefix each statement with a **Comment.**

Item Id	Author Id	Year	Week	Hours	*Reason for failure*
101	48	2012	1	1	*bid/authid combination does NOT exist in parent*
109	42	2012	2	2	*bid/authid combination does NOT exist in parent*
101	42	2014	9	6	*out of range workyear*
101	45	2012	55	3	*out of range workweek*
108	40	2012	7	120	*out of range workhours*

2D. SQL Queries

Write the following queries and add each to the .TXT file. Prefix each statement with an appropriate comment.

<table>
<tr><th>Query</th><th>Requirement</th><th>Sample Output</th></tr>
<tr><td>1</td><td>Write a query that lists rows from the work table.
Columns:
• author id
• work year
• work week
• work hours
Sequence:
• author id
• work year
• work week</td><td>
<table>
<tr><th>authid</th><th>workyear</th><th>workweek</th><th>workhours</th></tr>
<tr><td>40</td><td>2011</td><td>52</td><td>4</td></tr>
<tr><td>40</td><td>2012</td><td>4</td><td>15</td></tr>
<tr><td>40</td><td>2012</td><td>6</td><td>6</td></tr>
<tr><td>42</td><td>2012</td><td>5</td><td>6</td></tr>
<tr><td>42</td><td>2012</td><td>5</td><td>5</td></tr>
<tr><td>42</td><td>2012</td><td>6</td><td>4</td></tr>
<tr><td>42</td><td>2012</td><td>7</td><td>5</td></tr>
<tr><td>45</td><td>2012</td><td>5</td><td>10</td></tr>
<tr><td>45</td><td>2012</td><td>7</td><td>10</td></tr>
<tr><td>47</td><td>2012</td><td>4</td><td>8</td></tr>
<tr><td>47</td><td>2012</td><td>5</td><td>5</td></tr>
<tr><td>47</td><td>2012</td><td>6</td><td>7</td></tr>
<tr><td>47</td><td>2012</td><td>6</td><td>5</td></tr>
<tr><td>47</td><td>2012</td><td>8</td><td>4</td></tr>
<tr><td>47</td><td>2012</td><td>8</td><td>8</td></tr>
<tr><td>48</td><td>2012</td><td>10</td><td>15</td></tr>
<tr><td>48</td><td>2012</td><td>11</td><td>4</td></tr>
<tr><td>48</td><td>2012</td><td>12</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>2</td><td>Write a query that sums the total number of hours worked by each author by each year.
Columns:
• author id
• work year
• total hours
Sequence:
• total hours descending
• author id</td><td>
<table>
<tr><th>authid</th><th>workyear</th><th>Total hours</th></tr>
<tr><td>47</td><td>2012</td><td>37</td></tr>
<tr><td>40</td><td>2012</td><td>21</td></tr>
<tr><td>42</td><td>2012</td><td>20</td></tr>
<tr><td>45</td><td>2012</td><td>20</td></tr>
<tr><td>48</td><td>2012</td><td>20</td></tr>
<tr><td>40</td><td>2011</td><td>4</td></tr>
</table>
</td></tr>
</table>

<table>
<tr>
<td>3</td>
<td>Write a query that lists rows from the work table.
Columns:
• book id
• author id
• work year
• work week
• work hours
Sequence:
• book id
• author id
• work year
• work week</td>
<td>Output need to determined by the group</td>
</tr>
<tr>
<td>4</td>
<td>Write a query that sums the total number of hours worked by each book by each author by each year.
Columns:
• book id
• author id
• work year
• total hours
Sequence:
• book id
• author id
• work year</td>
<td>
<table>
<tr><th>BID</th><th>authid</th><th>workyear</th><th>Total hours</th></tr>
<tr><td>101</td><td>42</td><td>2012</td><td>14</td></tr>
<tr><td>101</td><td>45</td><td>2012</td><td>20</td></tr>
<tr><td>105</td><td>42</td><td>2012</td><td>6</td></tr>
<tr><td>105</td><td>47</td><td>2012</td><td>23</td></tr>
<tr><td>108</td><td>40</td><td>2011</td><td>4</td></tr>
<tr><td>108</td><td>40</td><td>2012</td><td>21</td></tr>
<tr><td>108</td><td>47</td><td>2012</td><td>4</td></tr>
<tr><td>109</td><td>47</td><td>2012</td><td>10</td></tr>
<tr><td>113</td><td>48</td><td>2012</td><td>20</td></tr>
</table>
</td>
</tr>
<tr>
<td>5</td>
<td>Copy and Modify above query

The 4th column must show the total hours worked multiplied by the pay rate for the author/book.</td>
<td>
<table>
<tr><th>BID</th><th>authid</th><th>workyear</th><th>Total Pay</th></tr>
<tr><td>101</td><td>42</td><td>2012</td><td>350</td></tr>
<tr><td>101</td><td>45</td><td>2012</td><td>640</td></tr>
<tr><td>105</td><td>42</td><td>2012</td><td>156</td></tr>
<tr><td>105</td><td>47</td><td>2012</td><td>575</td></tr>
<tr><td>108</td><td>40</td><td>2011</td><td>180</td></tr>
<tr><td>108</td><td>40</td><td>2012</td><td>945</td></tr>
<tr><td>108</td><td>47</td><td>2012</td><td>140</td></tr>
<tr><td>109</td><td>47</td><td>2012</td><td>420</td></tr>
<tr><td>113</td><td>48</td><td>2012</td><td>800</td></tr>
</table>
</td>
</tr>
</table>

Task 3:

Write a series of **SQL** statements that **DROP** all tables that were created in Tasks above. (NOTE: You must determine the **correct** sequence in which to drop these tables). Each statement must end with a semi-colon.

E.g.:
```
DROP TABLE BRANCH;
DROP TABLE EMPLOYEE ;
```

SQL script Instructions

Your SQL statement **must** follow these rules:

a. Each SQL statement **must** end with a **semi colon;**

b. Each SQL statement must be preceded by a **comment**
(Except Insert statements where only 1 statement is required for where multiple rows are inserted in to the same table)

c. Each SQL Keyword in a Query (SELECT, FROM, WHERE etc) must be at the beginning of a new line

--Example 1
Filename:
SAMPLE_1.TXT:

```
/* 17385606 Chandrika Kumaratunga
   12435689 Mahinda Rajapaksa */
-- Create the employee table;
CREATE TABLE employee (
empid               NUMBER,
empname VARCHAR2(30) PRIMARY KEY );
-- Insert data into employee table;
INSERT INTO employee (empid, empname) VALUES (1, 'Olga');
INSERT INTO employee (empid, empname) VALUES (2, 'Boris');
-- Query 1a;
SELECT              empid,empname
FROM                employee
WHERE               empid >= 1
AND                 empid <=5
ORDER BY 2;
```