



MONASH University

Office Use Only

--	--	--

**Semester Two 2022
Examination Period**

Faculty of Information Technology

Sample Exam

EXAM CODES:

FIT3003

TITLE OF PAPER:

Business Intelligence and Data Warehousing - SAMPLE 1

EXAM DURATION:

2 hours 10 minutes or 130 minutes

THIS PAPER IS FOR STUDENTS STUDYING AT: (tick where applicable)

- ☐ Caulfield ☒ Clayton ☐ Parkville ☐ Peninsula
☐ Monash Extension ☐ Off Campus Learning ☒ Malaysia ☐ Sth Africa
☐ Other (specify)

During an exam, you must not have in your possession any item/material that has not been authorised for your exam. This includes books, notes, paper, electronic device/s, mobile phone, smart watch/device, calculator, pencil case, or writing on any part of your body. Any authorised items are listed below. Items/materials on your desk, chair, in your clothing or otherwise on your person will be deemed to be in your possession.

No examination materials are to be removed from the room. This includes retaining, copying, memorising or noting down content of exam material for personal use or to share with any other person by any means following your exam.

Failure to comply with the above instructions, or attempting to cheat or cheating in an exam is a discipline offence under Part 7 of the Monash University (Council) Regulations, or a breach of instructions under Part 3 of the Monash University (Academic Board) Regulations.

AUTHORISED MATERIALS

OPEN BOOK

☐ YES

☒ NO

CALCULATORS

☐ YES

☒ NO

SPECIFICALLY PERMITTED ITEMS

☐ YES

☒ NO

if yes, items permitted are:

Candidates must complete this section if required to write answers within this paper

STUDENT ID: _____

DESK NUMBER: _____

Question 1:

Monash International would like to analyse their policy in regard to English requirement for admission into a course. Monash International has the following data:

Table: Student_IELTS

Student ID	Student Name	Listening	Reading	Writing	Speaking	Overall
228493	Sooying Tan	6.5	6.5	6.0	7.0	6.5
229094	Xuebing Lu	5.5	5.5	5.5	5.5	5.5
231289	Amandh Kumar	6.0	7.0	6.0	7.0	6.5
234354	Agus Hidayat	5.5	6.0	6.0	6.5	6.0
234355	Budi Rahayu	7.0	7.0	7.0	7.0	7.0
...						
...						

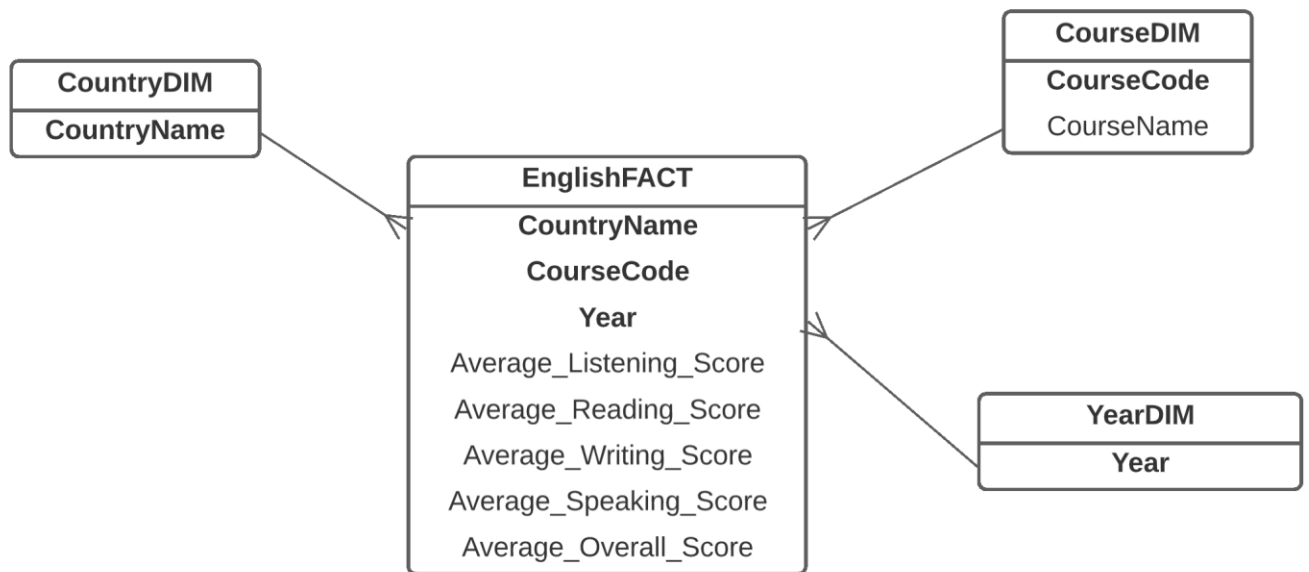
Table: Student_Course

Student ID	Student Name	Course	StarYear
228493	Sooying Tan	MBIS	2013
229094	Xuebing Lu	MBIS	2013
231289	Amandh Kumar	MIT	2013
234354	Agus Hidayat	MIT	2013
234355	Budi Rahayu	MIT	2013
...			
...			

Table: Student

Student ID	Student Name	Address	Suburb	Phone Number	Country
228493	Sooying Tan				Singapore
229094	Xuebing Lu				China
231289	Amandh Kumar				India
234354	Agus Hidayat				Indonesia
234355	Budi Rahayu				Indonesia
...					
...					

A data warehouse based on the above data has been created. A star schema is shown as follows:



Questions:

- The above star schema will not produce a correct analysis of the fact measures. Explain why. Explain your answer using more concrete examples or data. **(5 marks)**
- How do you correct this problem by changing the fact measures of the above star schema? Explain your solution using more concrete examples or data. **(5 marks)**

Write your answer here:

Question-a

If the operational database contains 10 students from Indonesia doing MIT in 2013, the fact table will contain only one entry, which aggregates these 10 students. The fact will aggregate the IELTS scores (i.e. listening, reading, writing, speaking, and overall score) and get the average (e.g. average listening, average reading, average writing, average speaking, and average overall score). This will be done for each of combination of country, course, and year.

However, if in the OLAP, we would like to get the average scores of students from a specific country, say Indonesia, then we will need to get all the records where country is Indonesia from the fact table. And then average it again.

The records in the fact table are already averaged. If we average on top of another average, the result will be incorrect.

Continue your answer here:

For example, the fact table of the above sample data is shown as follows:

Table: Fact

CountryName	CourseCode	Year	AvgList	AvgRead	AvgWrit	AvgSpeak	AvgOvrall
China	MBIS	2013	5.5	5.5	5.5	5.5	5.5
India	MIT	2013	6.0	7.0	6.0	7.0	6.5
Indonesia	MIT	2013	6.25	6.5	6.5	6.75	6.5
Singapore	MBIS	2013	6.5	6.5	6.0	7.0	6.5

If we query the **Average Listening Score of MIT** students, then based on the above fact table, the average listening score is $(6.0+6.25)/2 = 6.125$.

Supposed in the operational database, there are three students from India and Indonesia (e.g. Amandh from India, and Agus and Budi from Indonesia):

Table: Student_IELTS

Student ID	Student Name	Listening	Reading	Writing	Speaking	Overall
231289	Amandh Kumar	6.0	7.0	6.0	7.0	6.5
234354	Agus Hidayat	5.5	6.0	6.0	6.5	6.0
234355	Budi Rahayu	7.0	7.0	7.0	7.0	7.0

If we look at these three students from India and Indonesia (Amand, Agus, and Budi), the average listening score is $(6.0+5.5+7.0)/3 = 6.17$.

This example shows that keeping the average score in the fact will produce incorrect OLAP query results.

Question-b.

Hence, average is not a good fact measures. We should not have average_reading, average_listening, etc in the fact.

On the other hand, we should have total_reading, total_listening, total_speaking, total_writing, and total_overall in the fact. We also need total_number_of_students in the fact.

So, in OLAP, if we want to get the **Average Listening Score of MIT** course students, we will get the **total_listening** and **total_number_of_students**, and then divide them to get the average. Using the example above, **total_listening** = $(6.0+5.5+7.0) = 18.5$, and **total_number_of_students** = 3. Then listening average is $18.5/3 = 6.17$

Question 2:

Data cleaning is an important part in building a clean and correct data warehouse. Data cleaning is often needed, because there are mistakes and inconsistencies in the operational database. Before data cleaning is done, we need to do data exploration on the operational database in order to find out if there are any mistakes and inconsistencies in the operational database.

The following are the four tables in the operational database:

```
SQL> desc dw.uselog;
```

Name	Null?	Type
LOG_DATE	NOT NULL	DATE
LOG_TIME	NOT NULL	DATE
STUDENT_ID	NOT NULL	CHAR(11)
ACT		CHAR(1)

```
SQL> desc dw.student;
```

Name	Null?	Type
SEX		VARCHAR2(2)
FULL_PART		VARCHAR2(2)
TYPE		VARCHAR2(4)
CLASS_ID		VARCHAR2(6)
MAJOR_CODE		VARCHAR2(8)
STUDENT_ID	NOT NULL	CHAR(11)

```
SQL> desc dw.class;
```

Name	Null?	Type
CLASS_DESCRIPTION		CHAR(50)
CLASS_ID		VARCHAR2(6)

```
SQL> desc dw.major;
```

Name	Null?	Type
MAJOR_NAME		CHAR(35)
MAJOR_CODE		VARCHAR2(8)

Questions:

- Write the SQL command to find out if there are duplicate student records (**5 marks**)
- Write the SQL command to find out if there are records in dw.uselog whereby the Student_ID exists in dw.uselog actually do not exist in dw.student (**5 marks**)

Write your answers here:

(a) 5 marks (group by and count = 2 marks, having = 3 marks)

```
select student_id, count(*)
from dw.student
group by student_id
having count(*) > 1;
```

Notes for studying: students need to understand the difference between WHERE and HAVING

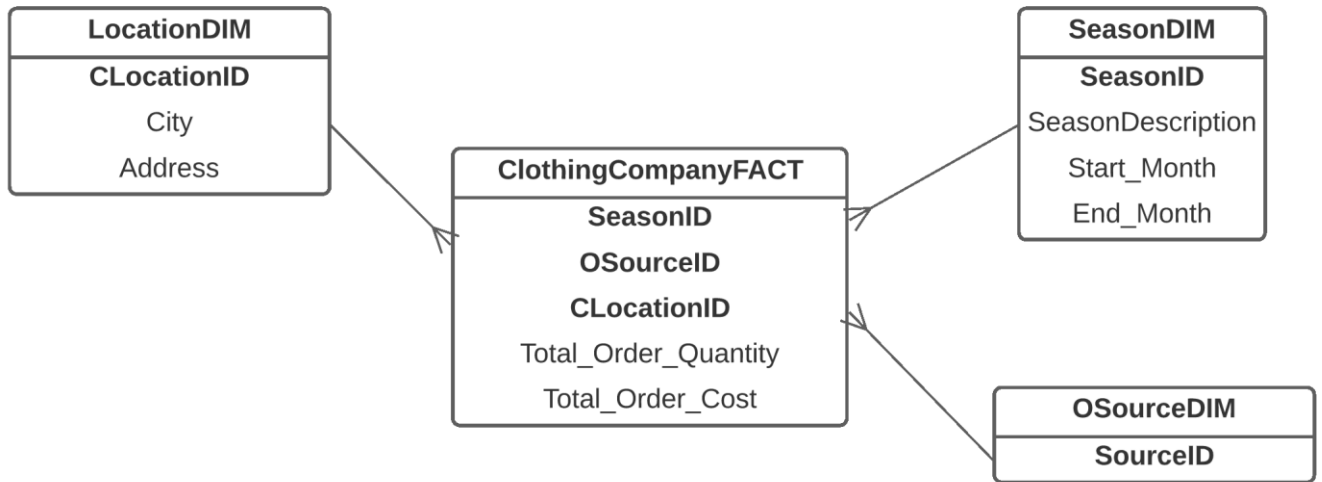
(b) 5 marks (not in = 3 marks, subquery = 2 marks)

```
select *
from dw.uselog
where student_id NOT IN
    (select student_id from dw.student);
```

Notes for studying: students need to understand nested queries

Question 3:

Given the following schema:



The tables (e.g. ClothingCompany fact and the three dimensions) have been created and populated with an adequate number of records.

The table names and attributes are shown in the above star schema. In the Fact table, the total order quantity and total order cost attributes are included.

Write the SQL for the following advanced OLAP queries:

- Perform a **CUBE** operation (use all dimensions). Display each TotalOrderCost and the subtotals. **(3 marks)**
- Like question (a) above, but now perform a **ROLLUP** operation. **(3 marks)**
- Perform a **CUMMULATIVE SUM** of the TotalOrderCost of all WEBSITE orders (use all dimensions). **(4 marks)**
- Like question (c) above, perform a CUMMULATIVE SUM of the TotalOrderCost but **PARTITIONED** based on the OSourceID, that is one partition for Phone orders, one partition for Fax orders, and one partition for Website orders. **(4 marks)**
- Show the total order costs of each source order, and **RANK** them. **(3 marks)**
- Display the source order that generates the highest total order cost. **(3 marks)**

Write your answer here:

a)

```
Select l.city, s.S_Desc, o.sourceid, sum(c.TotalOrderCost) as OrderCost
from LocationDim l, SeasonDim s, OsourceDim o, ClothingCompanyFact c
where l.CLocationID = c.CLocationID
and s.SeasonID = c.SeasonID
and o.sourceid = c.sourceID
group by cube (l.city, s.S_Desc, o.sourceid);
```

Notes:

1. In the above SQL, assume we use l.city (from LocationDIM), and s.s_desc (from SeasonDIM)
2. Alternatively, because OSourceDIM has only one attribute sourceID, we can use sourceID attribute in the fact. In the real life, OSourceDIM has many more attributes, and therefore, a join with the OSourceDIM table is necessary
3. You can use DECODE and GROUPING for better formatting

```
Select l.city, s.S_Desc, c.sourceid, sum(c.TotalOrderCost) as OrderCost
from LocationDim l, SeasonDim s, ClothingCompanyFact c
where l.CLocationID = c.CLocationID
and s.SeasonID = c.SeasonID
group by cube (l.city, s.S_Desc, c.sourceid);
```

b)

same as (a), but change the cube with rollup

Continue your answers here:

```
c)
Select l.city, s.S_Desc, sum(c.TotalOrderCost),
      TO_CHAR(SUM(SUM(c.TotalOrderCost))
      OVER(ORDER BY l.city, s.S_Desc ROWS UNBOUNDED PRECEDING),
      '9,999,999.99') AS Cumulative_Total_Order_Cost
from LocationDim l, SeasonDim s, OsourceDim o, ClothingCompanyFact c
where l.CLocationID = c.CLocationID
and s.SeasonID = c.SeasonID
and o.sourceid = c.sourceID
and o.SourceID = 'WEBSITE'
group by (l.city, s.S_Desc);
```

Notes:

1. The above will produce a listing sorted by City and then by Season (e.g. seasons within the same city will be sorted according to the season). The cumulative will be based on this ordering. This will only work if the query has ONE city. If it has multiple cities, it will not make sense that the cumulative total order cost for the second city will start from the first city the first season. Hence, the above query should be like this (e.g. to limit to one city only, such as Melbourne):

```
Select l.city, s.S_Desc, sum(c.TotalOrderCost),
      TO_CHAR(SUM(SUM(c.TotalOrderCost))
      OVER(ORDER BY l.city, s.S_Desc ROWS UNBOUNDED PRECEDING),
      '9,999,999.99') AS Cumulative_Total_Order_Cost
from LocationDim l, SeasonDim s, OsourceDim o, ClothingCompanyFact c
where l.CLocationID = c.CLocationID
and s.SeasonID = c.SeasonID
and o.sourceid = c.sourceID
and o.SourceID = 'WEBSITE' and
l.city = 'MELBOURNE' group by
(l.city, s.S_Desc);
```

2. If we would like to include multiple cities in the report, the cumulative needs to have a PARTITION, based on city. Hence, for each city, there will be a separate cumulative:

```
Select l.city, s.S_Desc, sum(c.TotalOrderCost),
      TO_CHAR(SUM(SUM(c.TotalOrderCost)) OVER(PARTITION
      BY l.city
      ORDER BY l.city, s.S_Desc ROWS UNBOUNDED PRECEDING),
      '9,999,999.99') AS Cumulative_Total_Order_Cost
from LocationDim l, SeasonDim s, OsourceDim o, ClothingCompanyFact c
where l.CLocationID = c.CLocationID
and s.SeasonID = c.SeasonID and
o.sourceid = c.sourceID and
o.SourceID = 'WEBSITE' group by
(l.city, s.S_Desc);
```

Continue your answer here:

d) Note that we assume that we limit the query to ONE city only:

```
Select o.SourceID, l.city, s.S_Desc, sum(c.TotalOrderCost),
       TO_CHAR(SUM(SUM(c.TotalOrderCost))
       OVER(PARTITION BY o.SourceID
            ORDER BY o.SourceID, l.city, s.S_Desc ROWS UNBOUNDED PRECEDING),
            '9,999,999.99') AS Cumulative_Total_Order_Cost
from LocationDim l, SeasonDim s, OsourceDim o, ClothingCompanyFact c
where l.CLocationID = c.CLocationID
and s.SeasonID = c.SeasonID
and o.sourceid = c.sourceID
and l.city = 'MELBOURNE'
group by (O.SourceID, l.city, s.S_Desc);
```

Notes:

1. The above will produce a listing sorted by SourceID, and then City and then by Season. In this case, there is only one City, which is Melbourne.
2. Each SourceID will have it's own cumulative.
3. If you have multiple cities, you could have a separate cumulative for SourceID combined with City. It means SourceID 'Phone' and City 'Melbourne' will have one set of cumulative, whereas SourceID 'Phone' and City 'Sydney' will have a separate set of cumulative:

```
Select o.SourceID, l.city, s.S_Desc, sum(c.TotalOrderCost),
       TO_CHAR(SUM(SUM(c.TotalOrderCost))
       OVER(PARTITION BY o.SourceID, l.City
            ORDER BY o.SourceID, l.city, s.S_Desc ROWS UNBOUNDED PRECEDING),
            '9,999,999.99') AS Cumulative_Total_Order_Cost
from LocationDim l, SeasonDim s, OsourceDim o, ClothingCompanyFact c
where l.CLocationID = c.CLocationID
and s.SeasonID = c.SeasonID and
o.sourceid = c.sourceID
group by (O.SourceID, l.city, s.S_Desc);
```

Continue your answer here:

e)

```
select OsourceID,  
       sum(TotalOrderCost) as OrderCost,  
       rank() OVER (ORDER BY sum(TotalOrderCost) DESC) from  
ClothingCompanyFact c  
group by OsourceID;
```

f)

```
SELECT *  
FROM (  
    select OsourceID, sum(TotalOrderCost) as OrderCost,  
           rank() OVER (ORDER BY sum(TotalOrderCost) DESC) as orderrank from  
    ClothingCompanyFact c  
    group by OsourceID;  
)  
WHERE orderrank <=1;
```

Notes:

1. Using MAX will be incorrect:

```
select OsourceID, max(TotalOrderCost) from  
ClothingCompanyFact;
```

As you cannot mix an attribute (e.g. OSourceID) and an aggregation function (e.g. max) in one Select statement.

2. You can do this, but it will only show the maximum TotalOrderCost without the OSourceID:

```
select max(TotalOrderCost) from  
ClothingCompanyFact;
```

3. You cannot do a group by either:

```
select OsourceID, max(TotalOrderCost) from  
ClothingCompanyFact  
group by OSourceID;
```

because it will get the maximum TotalOrderCost for each sourceID, which is not what the query asks.

Question 4:

There is a toll way (or toll road) in a metropolitan city (such as CityLink or EastLink in Melbourne, or any similar toll roads in other major cities in the world). This toll way has a number of gates, where the motorist needs to pay. Every time a motorist passes through this toll gate, the registration number of the vehicle, vehicle type (e.g. car, bus, truck, etc), amount paid, and time, are recorded in the operational database.

A data warehouse needs to be built, for analysing the *revenue* from the toll payments. The management would like to drill down this revenue based on the *tollgate* (there is a number of toll gates along the toll way), *day of week* (e.g. weekdays, weekends), and *time period of a day* (e.g. peak hours, non-peak hours, late nights).

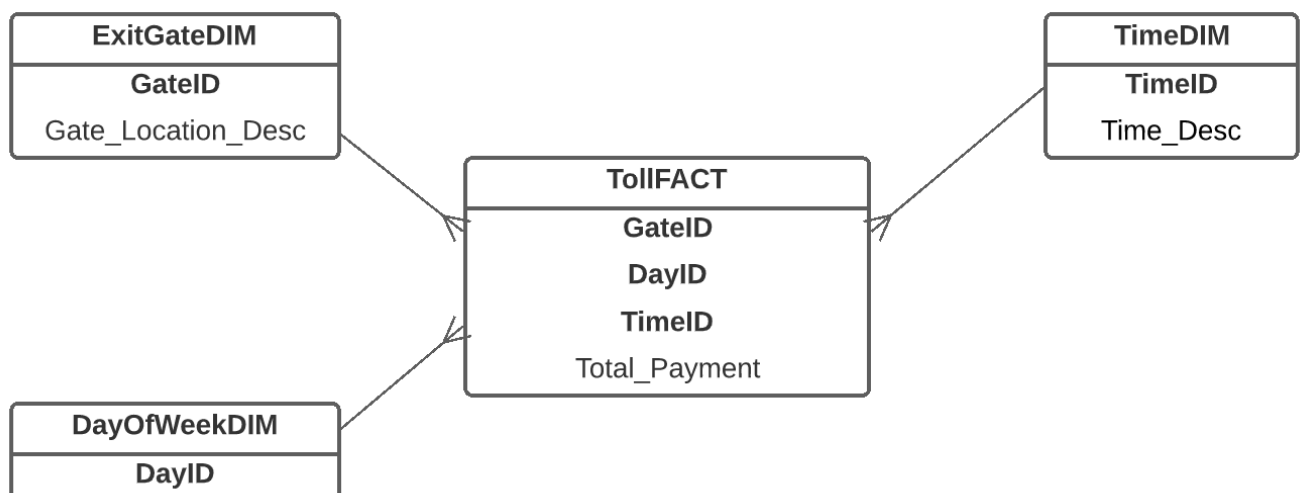
You are required to draw **three levels of star schemas** showing three different levels of aggregation for the above data warehouse. You also need to explain each of the three star schemas, by contrasting the level of aggregation. Level-0 star schema contains the most detailed data, whereas level-2 star schema is the highly aggregated (e.g. containing highly aggregated data).

Questions:

- (a) Draw a level-2 star schema and explain why it is a level-2 schema **(6 marks)**
- (b) Draw a level-1 star schema and explain why it is a level-1 schema. You may want to add a new dimension, called *vehicle* (e.g. cars, trucks, busses, etc). You need to also explain the difference between level-1 and level-2 schemas. **(6 marks)**
- (c) Draw a level-0 star schema and explain why it is a level-0 schema. You also need to explain the difference between level-1 and level 0 schemas. **(8 marks)**

Write your answer here:

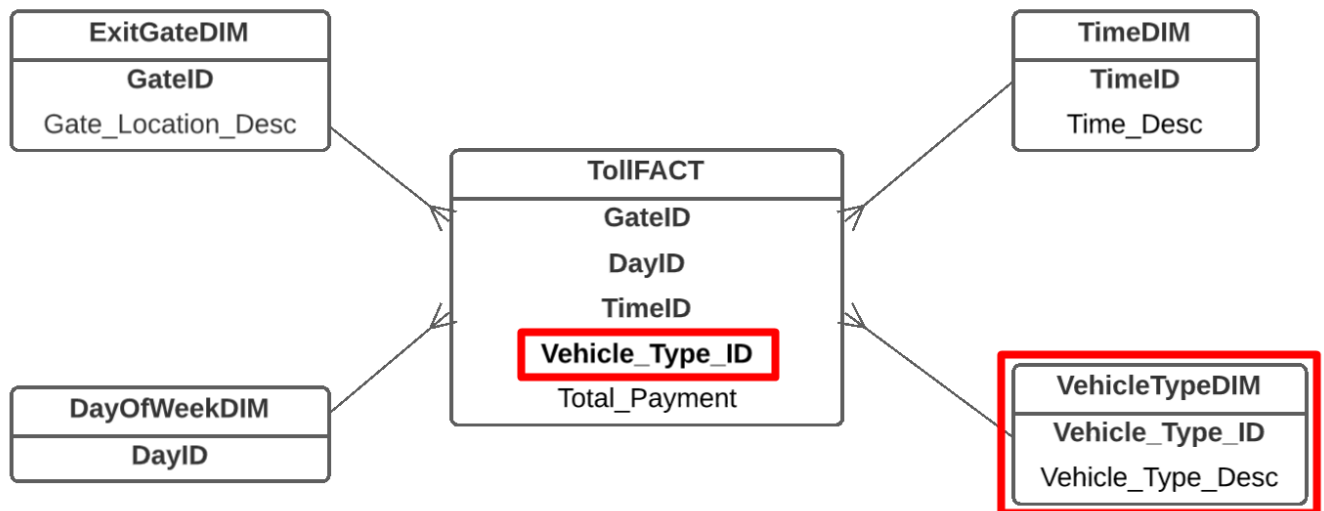
Level 2 – Highly aggregated. This is the most highly aggregated data where we keep “Total Payment” in the Fact table, and the Dimensions are based on Days of Week (not the actual date), and Time Period (not the actual travel time).



Continue your answer here:

Level 1– We add with an additional dimension, for example the Vehicle Type Dimension. In the Vehicle Type Dimension, we store the vehicle types, such as bus, truck, cars, etc. Note that we do not keep the “Registration Number” of the car. The granularity is at the vehicle type level, not at an individual car level.

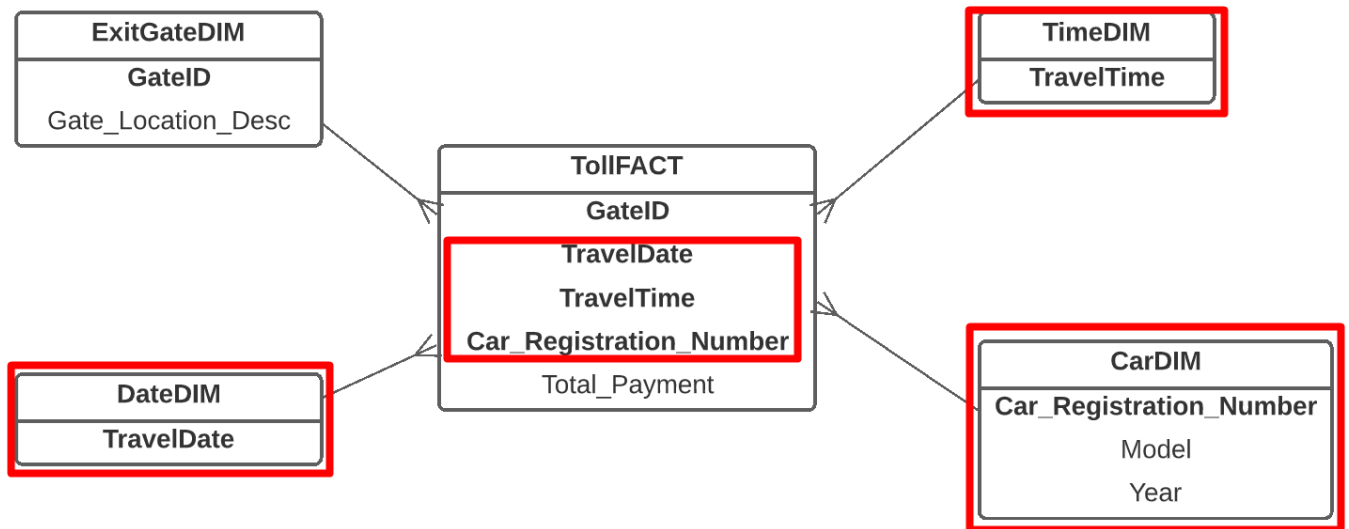
So, from level 2 to level 1, the total payment in level 2 is broken down by Vehicle Type. Other dimensions remain unchanged.



Continue your answer here:

Level 0– Detail Level

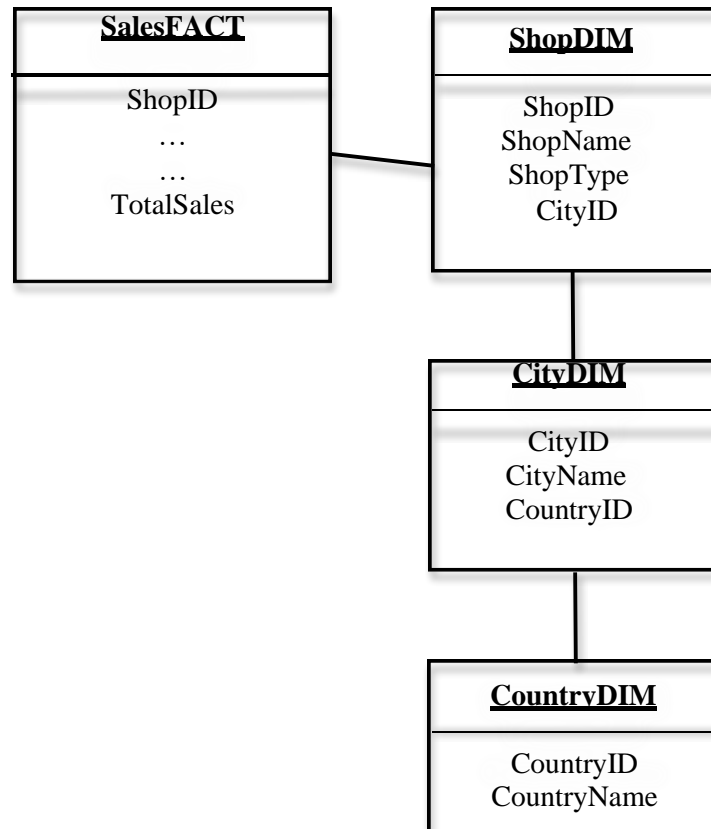
This is the most detail level as we have Date_DIM and Time_DIM to indicate the actual travel dates and travel time; as well as Car DIM to indicate the actual car (rather than just the type of vehicle)



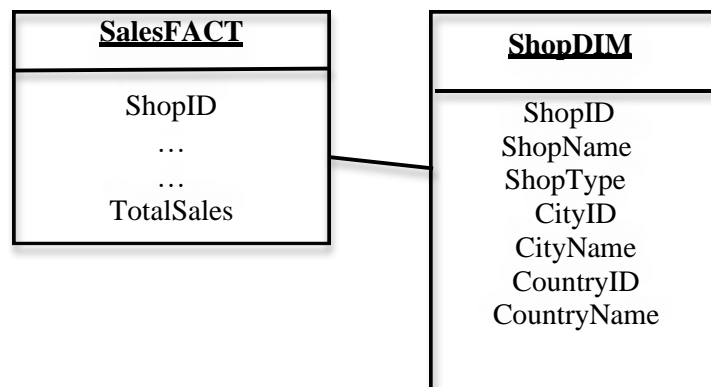
Question 5:

Consider the following star schemas. Star Schema-1 contains a hierarchy in the dimension, whereas Star Schema-2 collapses the hierarchy into one dimension.

Star Schema-1 (with hierarchy):



Star Schema-2 (without hierarchy):



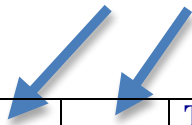
Questions:

- Draw sample table contents of the fact and dimension tables of the two star schemas. **(6 marks)**
- Compare and contrast the two star schemas using the sample tables in question (a) above. Explain the pros and cons of each star schema. **(4 marks)**

Write your answers here:

(a) Attributes from other dimensions
Star Schema-1

ShopFACT



ShopID			TotalSales
1	\$1,500,000
2	\$2,750,000
3	\$1,800,000

ShopDIM

ShopID	ShopName	ShopType	CityID
1	Myer	DepartmentStore	C
2	Coles	GroceryStore	C
3	BigW	DepartmentStore	M

CityDIM

CityID	CityName	CountryID
C	Canberra	AD
M	Melbourne	AD

CountryDIM

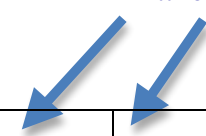
CountryID	CountryName
AD	Australia

Continue your answers here:

Star Schema-2

Attributes from other dimensions

ShopFACT



ShopID			TotalSales
1	\$1,500,000
2	\$2,750,000
3	\$1,800,000

ShopDIM

ShopID	ShopName	ShopType	CityID	CityName	CountryID	CountryName
1	Myer	DepartmentStore	C	Canberra	AD	Australia
2	Coles	GroceryStore	C	Canberra	AD	Australia
3	BigW	DepartmentStore	M	Melbourne	AD	Australia

(b)

Star Schema-1

Pros: normalized, minimized data duplication

Cons: When producing a report, we need to join the fact table with three dimension tables: shopDim, cityDim, and countryDim. Hence, we need three join operations.

Star Schema-2

Cons: unnormalized, has data duplication

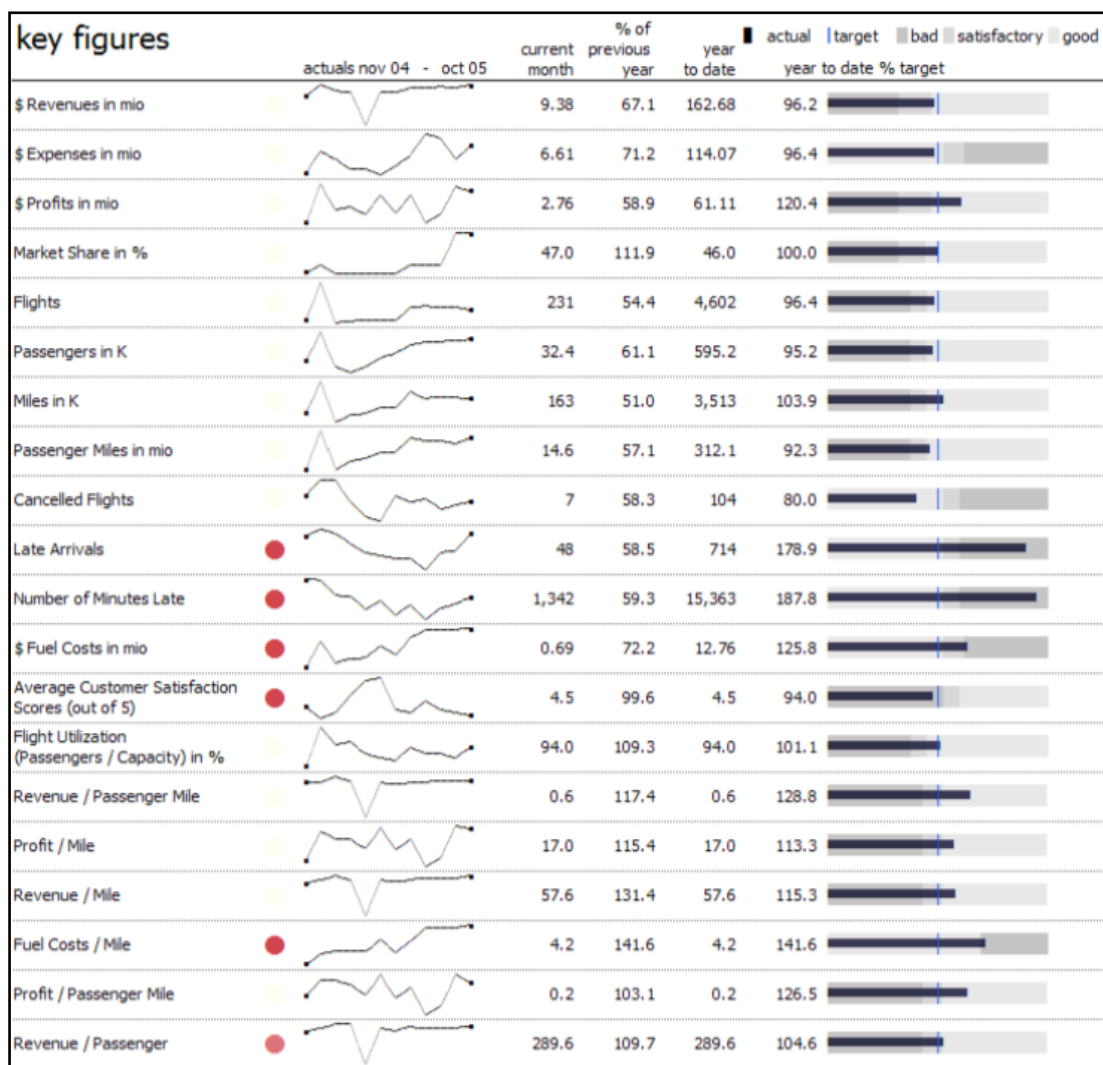
Pros: When producing a report, we need to join the fact table with shopDim only. Hence, we need one join operation only.

Question 6:

- a) Dashboards provide overview of a system in a visual and often interactive display, with the most important information about the system. It can be customised for specific groups of users, or for different purposes. There are basically three types of dashboard, namely strategic, analytic and operational.

The following figure shows a part of the dashboard in a BI system for an airline company. What type of dashboard is this? What type of users would use this dashboard? Explain your answer.

[5 marks]

**Sample Answer:**

The dashboard is of strategic type.

It contains important information, such as revenue, profit, flight status and etc. for one year period. The sparklines and progress bars provide high-level measures of past and forecast performance.

It can be used by managers to monitor their business health and progress towards meeting strategic objectives.

- b) Compared to dashboard, reports provide minimal interactivity and analysis. A good report should require minimal cognitive effort in order to understand the contents. There are a few types of navigational methods when viewing reports in BI systems.

Explain drill up and drill-through navigational methods using examples.

[5 marks]

Sample Answer:

- Drill-Up – Navigating to a report with less detail following the levels of a pre-defined dimension hierarchy.

Example: Daily sales → Monthly sales → Quarterly sales → Yearly sales

- Drill-Through – Navigating from a summary report to report that shows the detailed transactions that made up the summary.

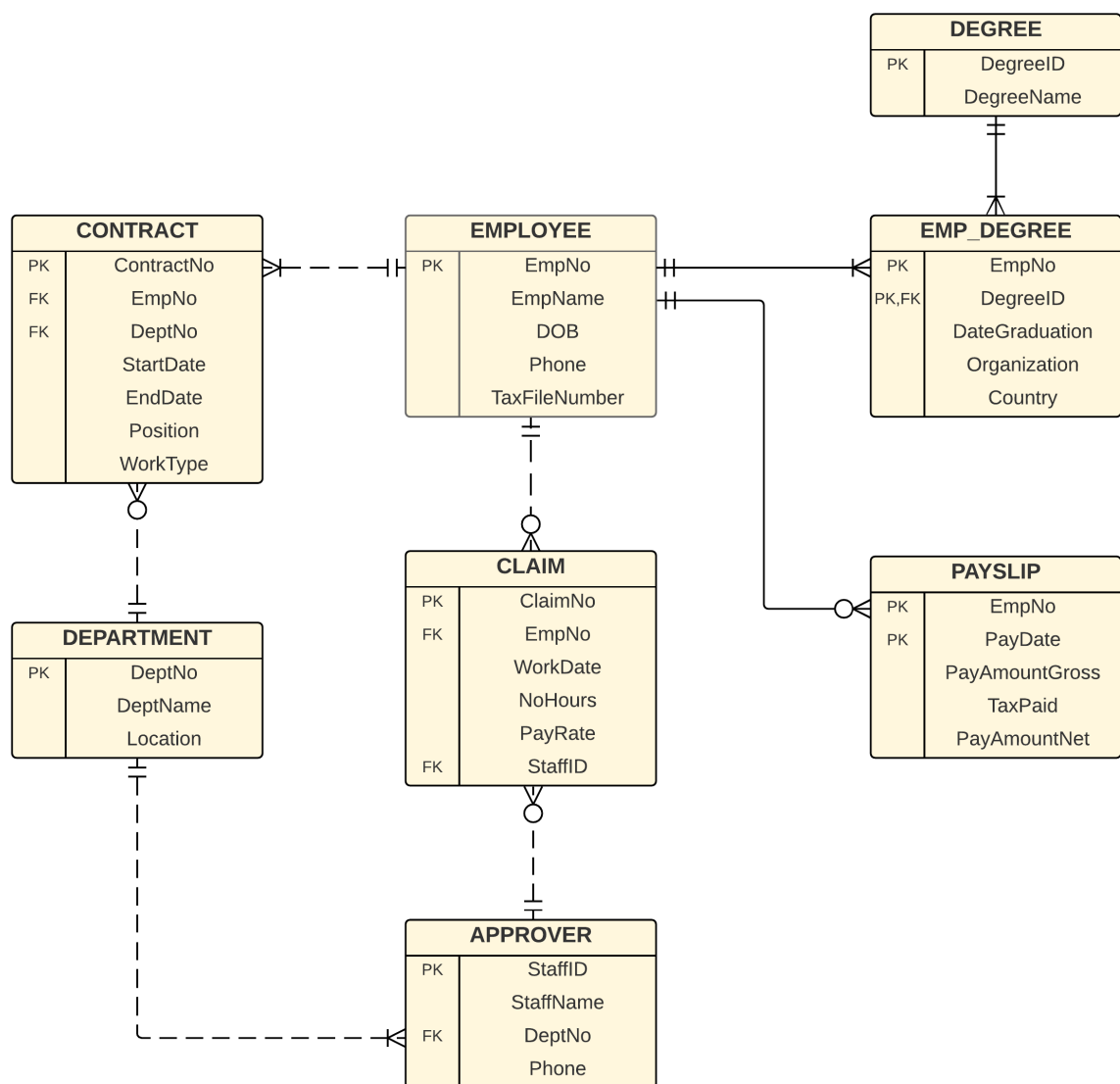
Example: From a summary sales report by country, click to see the detailed transactions within a specific country. The attributes between the summary and the detailed transactions are different.

Question 7

Monash University employs its students to do various jobs, such as tutoring, programming, etc. These jobs are called sessional jobs. For each sessional job, students need to sign a contract. For example, to do tutoring (one or more units), the student will sign a contract with Monash for one semester.

These sessional workers (e.g. sessional tutors) need to claim their work hours every week. This claim will need to be approved by a designated person in the faculty. Every fortnight, the sessional workers will get their pay.

The operational database, which keeps track of this system, is shown by the following E/R diagram.



You are required to build a data warehouse to analyse the following:

- The total number of contracts made every year.
- The total payment made to each employee in 2020.
- The yearly number of contracts made in each department.

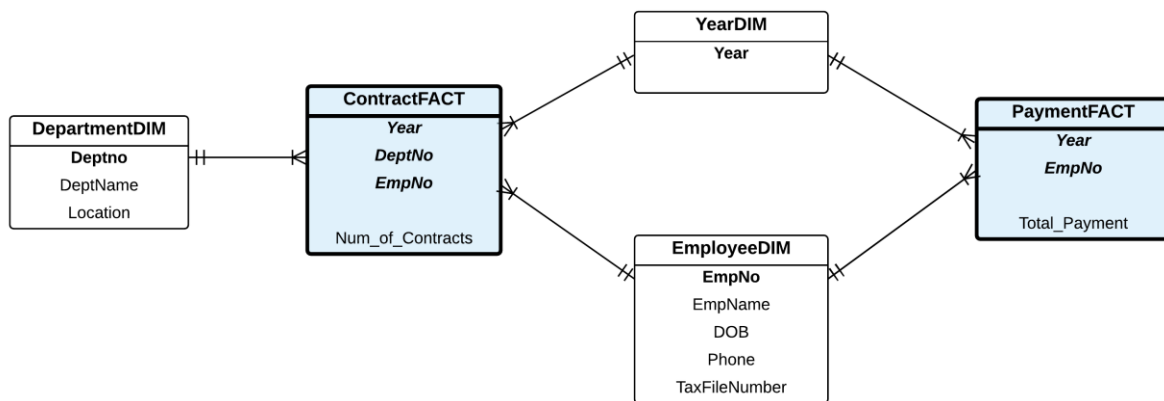
Question:

(a) Draw the star schema for this system. **(10 marks)**

(b) Write the SQL queries to create the fact and dimension tables. **(10 marks)**

Write your answers here:

(a) [Star Schema](#)



(b) SQL Queries

```
create table DepartmentDim as
select * from Department;
```

```
create table EmployeeDim as
select * from Employee;
```

```
create table YearDim as
select distinct to_char(StartDate, 'YYYY') as Year
from Contract
```

```
UNION
```

```
select distinct to_char(PayDate, 'YYYY') as Year
from Payslip;
```

```
create table ContractFact as
select
    to_char(C.StartDate, 'YYYY') as Year,
    E.EmpNo,
    D.DeptNo,
    COUNT(C.ContractNo) as Num_of_Contracts
from Employee E, Contract C, Department D
where E.EmpNo = C.EmpNo
and C.DeptNo = D.DeptNo
group by
    to_char(C.StartDate, 'YYYY'),
    E.EmpNo,
    C.DeptNo;
```

```
create table PaymentFact as
select
    to_char(P.PayDate, 'YYYY') as Year,
    E.EmpNo,
    SUM(P.PayAmountNet) as Total_Payment
from Employee E, Payslip P
where E.EmpNo = P.EmpNo
group by
    to_char(P.PayDate, 'YYYY'),
    E.EmpNo;
```

THE END