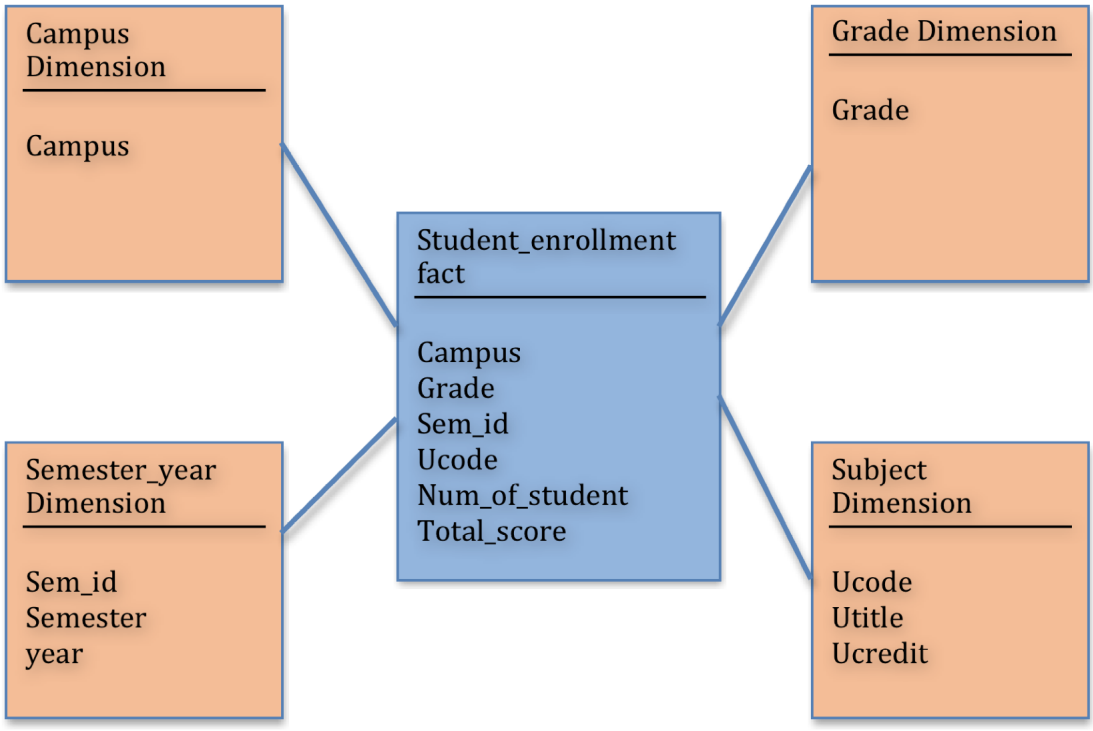
More on Case Study #1 (Student Enrollment)

1. Two dimensions vs. One dimension

The star schema of the first case study (Student-Enrollment system) is shown as follows.

Star Schema-1 (one semester-year dimension)



The SQL codes to create dimension and fact tables are as follows:

```
--Campus dimension
Create table campus_dim as
SELECT distinct Ocampus
FROM Offering;

--Semester_year dimension
Create table sem_year_dim as
SELECT distinct Oyear||Osem as sem_id, Oyear, Osem
FROM Offering;

--Subject Dimension
Create table subject_dim as
SELECT *
FROM subject;

--Grade Dimension
Create table grade_dim as
SELECT distinct Grade
FROM Enrollment;
```

```
--The Fact
CREATE TABLE student_enrollment_fact as
SELECT
   o.Ocampus,
   o.Oyear||o.Osem as sem_id,
   s.Ucode,
   e.Grade,
   count(st.sid) as num_of_student,
   sum(e.score) as Total_score
FROM subject s, enrollment e, offering o, student st
WHERE e.OID = o.OID
and s.Ucode = o.Ucode
and st.SID = e.SID
GROUP BY
   o.Ocampus,
   o.Oyear||o.Osem,
   s.Ucode,
   e.Grade;
```

The records in the data warehouse are as follows:

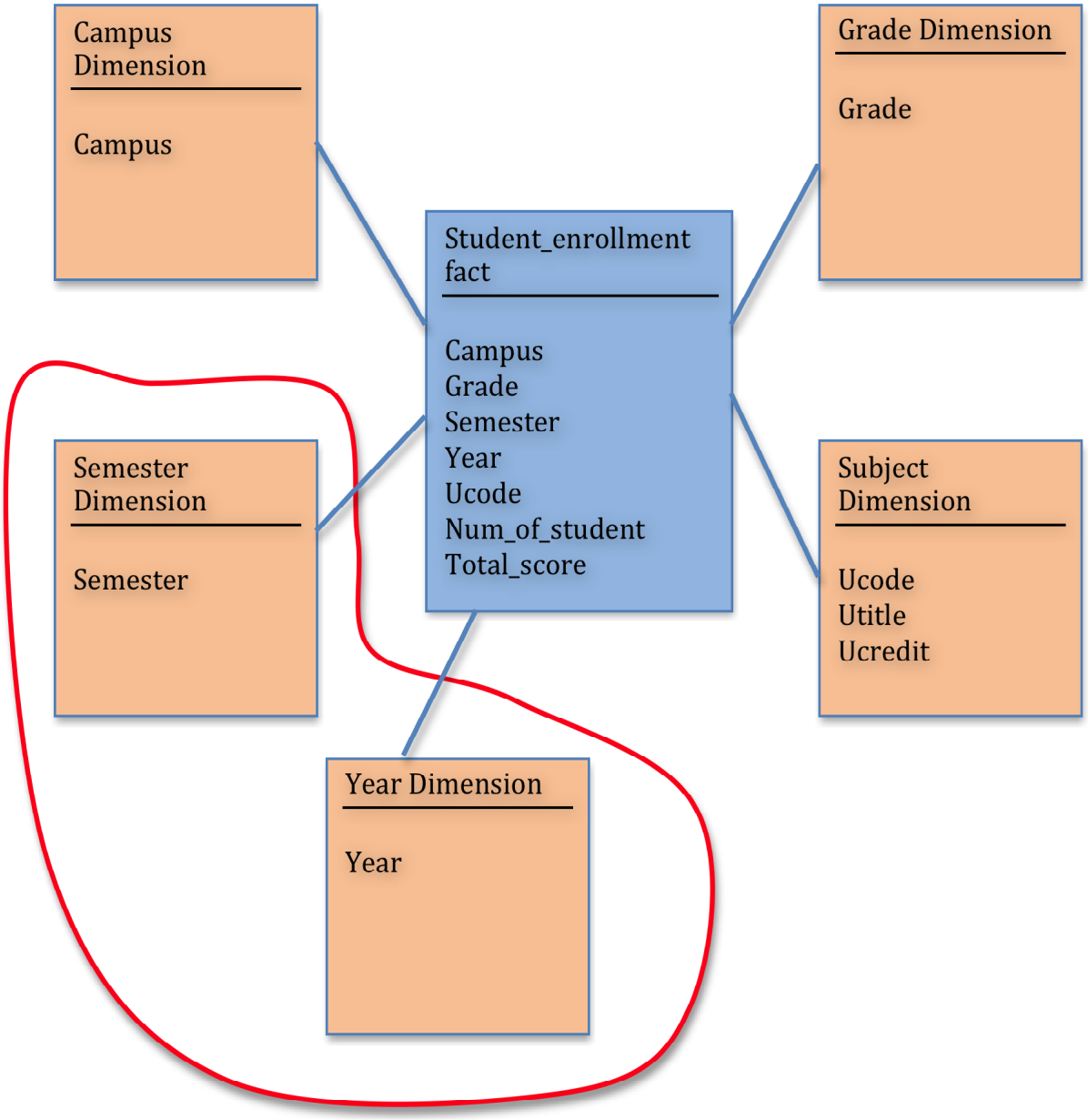
```
SQL> select * from campus_dim;

OCAMPUS
---------
City
Main
DE

SQL> select * from sem_year_dim;

SEM_ID     OYEAR      OSEM
---------- ---------- ----
20092      2009          2
20091      2009          1

SQL> select * from subject_dim;

UCODE      UTITLE                    UCREDIT
---------- -------------------- ----------
IT001      Database                       5
IT002      Java                           5
IT003      SAP                           10
IT004      Network                        5
IT005      ASP.NET                        5

SQL> select * from grade_dim;

GRADE
-----
HD
P
D
C
N
```

```
SQL> select * from student_enrollment_fact;

OCAMPUS    SEM_ID     UCODE      GRADE NUM_OF_STUDENT TOTAL_SCORE
---------- ---------- ---------- ----- -------------- -----------
Main       20091      IT001      HD                 3         253
Main       20091      IT001      D                  2         149
Main       20091      IT002      C                  1          65
Main       20091      IT002      N                  1          47
Main       20092      IT002      D                  2         151
City       20092      IT001      C                  1          64
Main       20091      IT001      N                  1          41
Main       20092      IT002      C                  1          64
DE         20092      IT004      P                  2         105
City       20091      IT003      C                  1          63
City       20092      IT001      N                  1          32

11 rows selected.
```

Look at particularly the `sem_year_dim` dimension, both in the SQL code and the contents of this dimension. In the above version of the star schema, the semester and the year information are actually combined into one dimension, called `sem_year_dim` dimension.

Can we have two separate dimensions: one for semester, and the other for year? The answer is “yes, you can”. The new star schema would look like as follows. Because the original sem_year_dim is now split into two dimensions: sem_dim, and year_dim, consequently, in the fact table, we will have two attributes: semester and year, instead of just one attribute sem_id.

Star Schema-2 (two separate dimensions for semester and year)



The SQL codes to create these two new dimensions (sem_dim and year_dim) and the fact tables are as follows:

```
--Semester dimension
Create table sem_dim as
SELECT distinct Osem as sem_id
FROM Offering;

--Year dimension
Create table year_dim as
SELECT distinct Oyear as year_id
FROM Offering;
```

```
--The Fact
CREATE TABLE student_enrollment_fact_2 as
SELECT
   o.Ocampus,
   o.Osem as sem_id,
   o.Oyear as year_id,
   s.Ucode,
   e.Grade,
   count(st.sid) as num_of_student,
   sum(e.score) as Total_score
FROM subject s, enrollment e, offering o, student st
WHERE e.OID = o.OID
and s.Ucode = o.Ucode
and st.SID = e.SID
GROUP BY
   o.Ocampus,
   o.Osem,
   o.Oyear,
   s.Ucode,
   e.Grade;
```

The contents of the two dimensions and the new fact are as follows:

```
SQL> select * from sem_dim;

    SEM_ID
----------
         1
         2

SQL> select * from year_dim;

   YEAR_ID
----------
      2009

SQL> select * from student_enrollment_fact_2;

OCAMPUS    SEM_ID  YEAR_ID UCODE      GR NUM_OF_STUDENT TOTAL_SCORE
--------- ------ -------- ---------- -- -------------- -----------
DE              2     2009 IT004      P               2         105
Main            1     2009 IT001      N               1          41
Main            1     2009 IT001      D               2         149
Main            2     2009 IT002      C               1          64
Main            1     2009 IT002      N               1          47
Main            2     2009 IT002      D               2         151
City            2     2009 IT001      N               1          32
Main            1     2009 IT002      C               1          65
City            1     2009 IT003      C               1          63
Main            1     2009 IT001      HD              3         253
City            2     2009 IT001      C               1          64

11 rows selected.
```

Now compare the contents of Fact-1 and Fact-2. They are almost the same, except that Fact-1 has one attribute from the sem_year_dim dimension, whereas Fact-2 has two attributes, called sem_id (from semester_dim) and year_id (from year_dim).

Conclusion: From the FACT point of view, it does not really matter whether we have two separate dimensions or just one dimension. They are the same. However, if two information, like semester and year, is often seen as one entity or one piece of information, it would be easier if the two information is present in one dimension.

2. Dimension Hierarchy (Hierarchy vs. No Hiearchy)

Suppose we have richer information about campus, as follows:

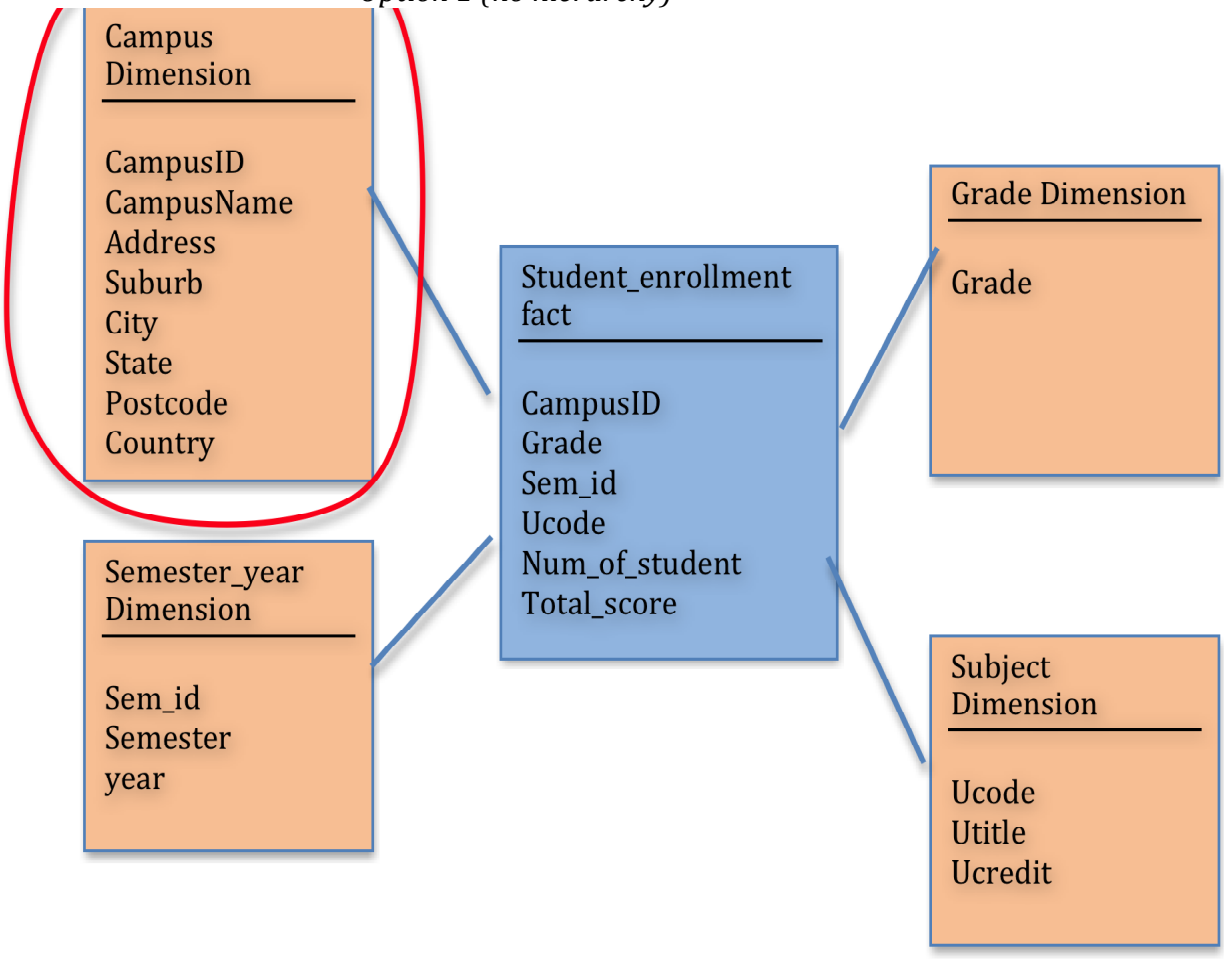
Campus Dimension

CampusID	CampusName	Address	Suburb	City	State	Postcode	Country
CL	Clayton Campus	Wellington Rd	Clayton	Melbourne	Victoria	3800	Australia
CA	Caulfield Campus	Dandenong Rd	Caulfield East	Melbourne	Victoria	3145	Australia
PA	Parkville Campus	Royal Parade	Parkville	Melbourne	Victoria	3052	Australia
SY	Sydney Campus	Opera Boulevard	Sydney	Sydney	New South Wales	2001	Australia
MA	Malaysia Campus	Jalan Lagoon	Bandar Sunway	Sydney	Selangor	47500	Malaysia
SA	South Africa Campus	Peter St	Johannesburg	Johannesburg	Johannesburg	1725	South Africa

In the star schema, we have two options:

Option 1 is to have one dimension called campusDIM containing all of these attributes.

Option 1 (no hierarchy)

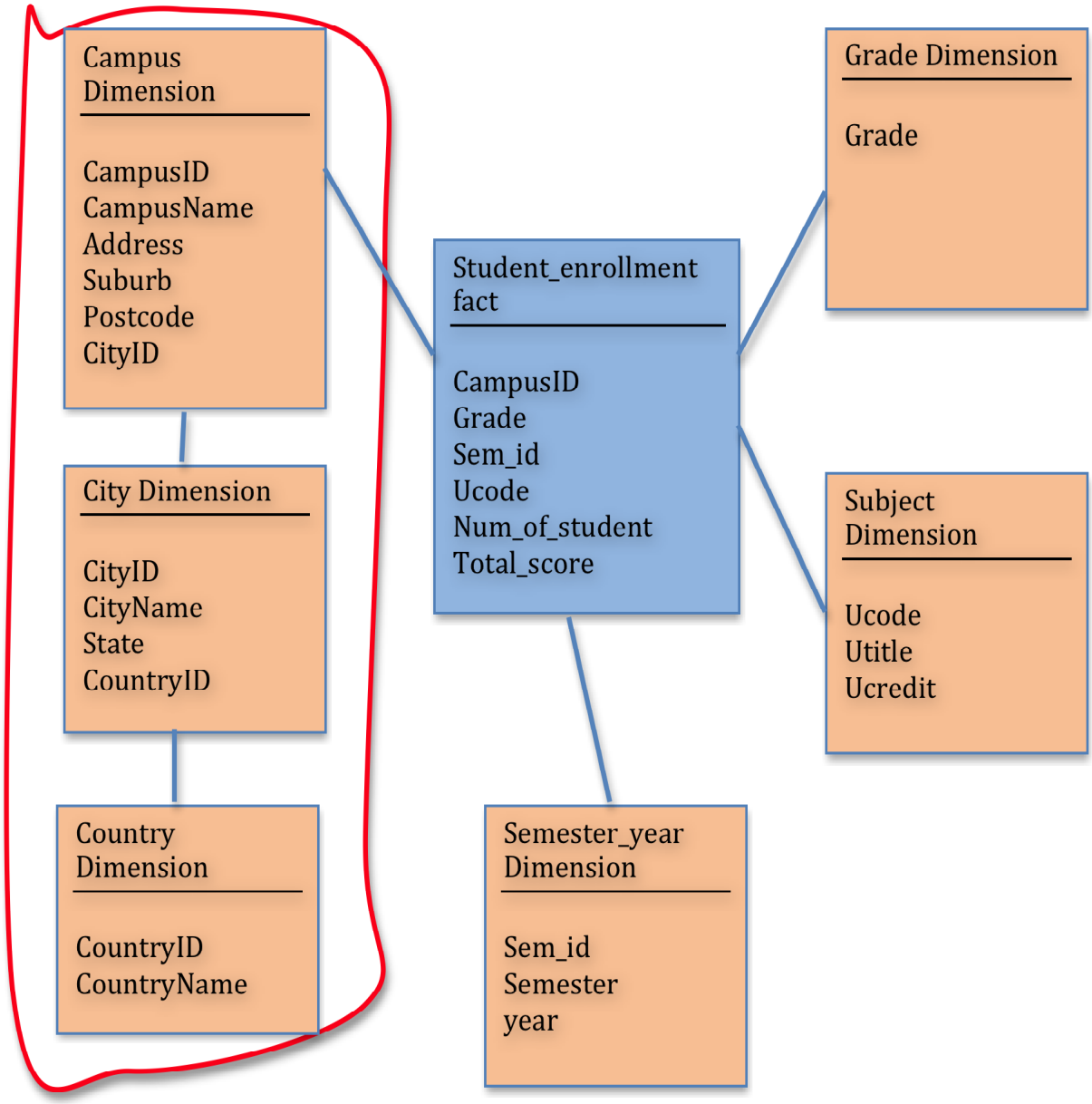


The SQL to create the Campus Dimension is shown as follows:

```
--Campus dimension
Create Table campus_dim As
SELECT Distinct Ocampus, <<and any other attributes, eg campus name, city, etc>>
FROM Offering;
```

Option 2 is to have a hierarchy of three dimensions (e.g. campusDIM connected to the Fact, campusDIM is also the parent of cityDIM – because campus-city forms a hierarchy, and cityDIM becomes the parent of countryDIM).

Option 2 (with hierarchy)



The Campus dimension is still linked to the Fact, whereas the City dimension is linked (through hierarchy) to the Campus, and the Country dimension is linked

to the City dimension. Therefore, there is a hierarchy from Campus to City, and to Country.

Note that the information about city is pulled out from the campus, and then create a hierarchy between campus and city. The information about country is also pulled out from the campus and city.

The SQL to create the three dimensions in the hierarchy is as follows:

```
--Campus dimension
Create Table campus_dim As
SELECT Distinct Ocampus, <<and any other attributes, eg campus name, etc>>
FROM <<table that stores information about campus, city, country>>;

--City dimension
Create Table city_dim As
SELECT Distinct Ocity, <<and any other attributes about the city>>
FROM <<table that stores information about campus, city, country>>;

--Country dimension
Create Table country_dim As
SELECT Distinct Ocountry, <<and any other attributes about the country>>
FROM <<table that stores information about campus, city, country>>;
```

The contents of the dimensions in the hierarchy (campus, city, and country dimensions are as follows):

Campus Dimension

CampusID	CampusName	Address	Suburb	Postcode	CityID
CL	Clayton Campus	Wellington Rd	Clayton	3800	MEL
CA	Caulfield Campus	Dandenong Rd	Caulfield East	3145	MEL
PA	Parkville Campus	Royal Parade	Parkville	3052	MEL
SY	Sydney Campus	Opera Boulevard	Sydney	2001	SYD
MA	Malaysia Campus	Jalan Lagoon	Bandar Sunway	47500	KUL
SA	South Africa Campus	Peter St	Johannesburg	1725	JNB

City Dimension

CityID	CityName	State	CountryID
MEL	Melbourne	Victoria	AU
SYD	Sydney	New South Wales	AU
KUL	Kuala Lumpur	Selangor	MA
JNB	Johannesburg	Johannesburg	SA

Country Dimension

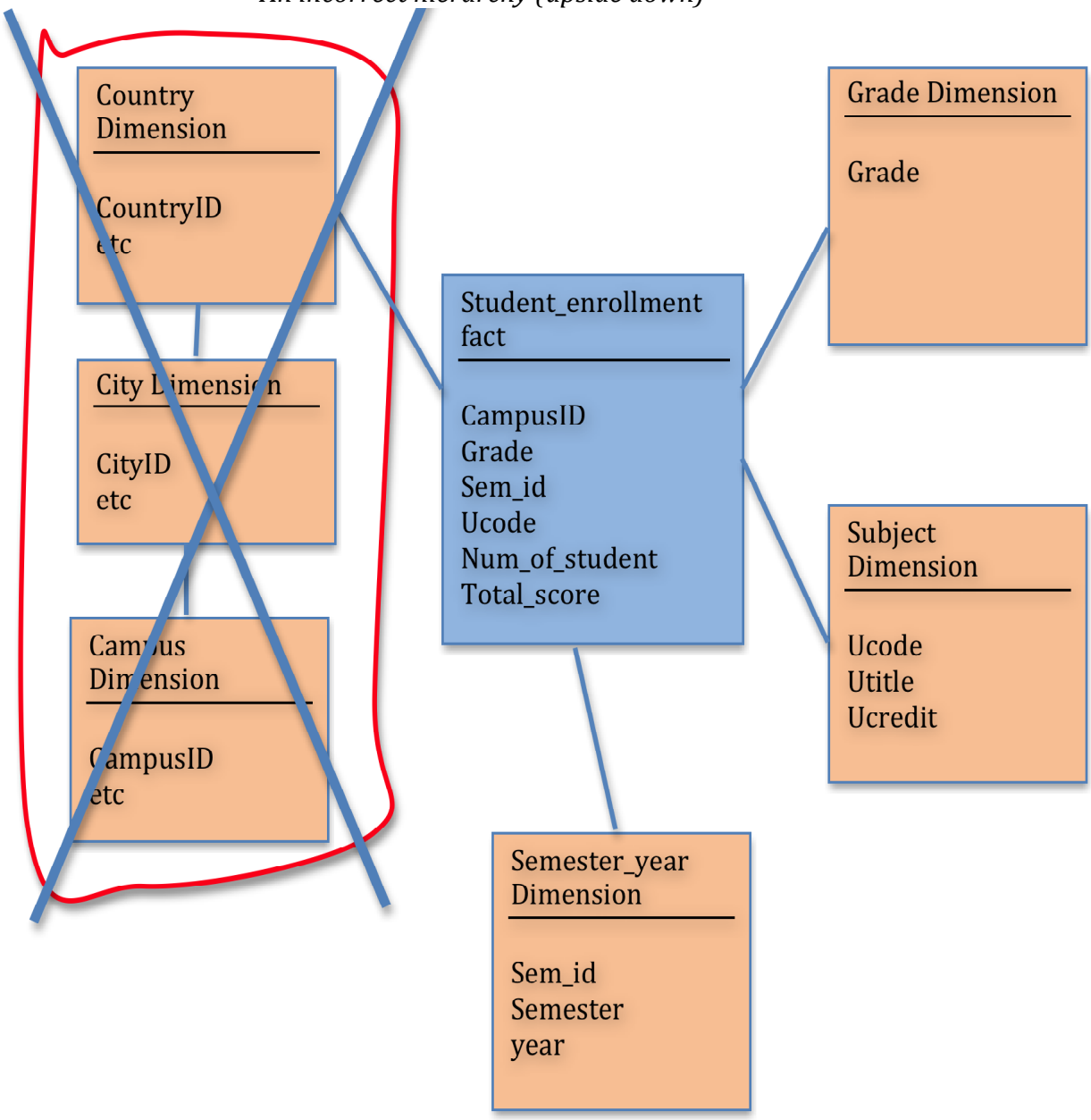
CountryID	CountryName
AU	Australia
MA	Malaysia
SA	South Africa

Note that the creation of the fact table is not affected at all. Either model (option 1 or option 2) will not affect the fact and other dimensions (e.g. grade, subject, semester_year dimensions).

Hierarchy should have the higher granularity or higher detail (e.g. campus) as a parent, which is attached to the fact. The lower granularity or lower detail (e.g. city) becomes the child in the hierarchy. The lowest granularity or the lowest detail (e.g. country) becomes the lowest in the hierarchy.

One of the most common mistakes is to wrong ordering of the dimensions in the hierarchy, such as the following star schema:

An incorrect hierarchy (upside down)



Hiearchy vs No Hierarchy

Since there are two options in implementing a star schema: with and without hierarchy, which option is more preferred?

1. *One table vs. many tables*
   Without hierarchy, there is only one table (e.g. the campus dimension table). With hierarchy, there are three tables (e.g. campus, city, and country dimension tables). The consequence is in the join query processing when producing reports.

   When you want to produce a report involving the fact and campus, with the no hierarchy option, we only need to use two tables: the campus dimension table, and the fact table. Hence, the join query processing is simple.

   With the hierarchy option, we need to join the fact with three tables (campus, city, and country). Consequently, the join query processing will be more complex

2. *Normalized vs. Unnormalized*
   With the hierarchy option, the tables are normalized, which follows the relational model. In contrast, the non-hierarchy option, the table (e.g. the campus dimension table) is unnormalized, which is basically 1NF with visible replication on information which is prone to anomalies (e.g. insert, update, and delete anomalies).

   Since no update, insert, or delete operations are performed to the dimension tables (once they have created), hence, anomalies become irrelevant. Therefore, the unnormalized option may be preferred, also due to a simpler join query processing.

3. *Drilling down and rolling up*
   People often mistakenly thought that the hierarchy model is good for drilling down. The hierarchy is not from country to city and to campus. The correct hierarchy is from the detail to the more general, such as from campus to city, and to country. Hence, we are not drilling down from campus to country, because drilling down should from something that is general (such as country) to something more detail (such as city and campus). In other words, the hierarchy model is not about drilling down information exploration.

   How about rolling up? Rolling up is exploring information from the detail to a more general. For example, retrieve number of students located at a campus level, and then rolling up to the country level.

   With the **hierarchy option**, we still need to do two queries: one for the campus and the second for the country.

```
Select CampusName, Sum(Num_of_student)
From Fact F, CampusDim C
Where F.CampusID = C.CampusID
Group By CampusName;

Select CountryName, Sum(Num_of_student)
From Fact F, CampusDim C1, CityDim C2, CountryDim C3
Where F.CampusID = C1.CampusID
And C1.CityID = C2.CityID
And C2.CountryID = C3.CountryID
Group By CountryName;
```

Whereas with the **non-hierarchy option**, we still need to do two queries:

```
Select CampusName, Sum(Num_of_student)
From Fact F, CampusDim C
Where F.CampusID = C.CampusID
Group By CampusName;

Select Country, Sum(Num_of_student)
From Fact F, CampusDim C
Where F.CampusID = C.CampusID
Group By Country;
```

From the query point of view, both need two queries.
From the query processing point of view, the non-hierarchy option uses one join operation only, because it only needs to join the fact and one dimension.
From the conceptual point of view, the hierarchy model does not actually offer a better roll up or drill down features.

If we insist to have a drill down, that is from country to campus, we still need to do the two queries above, but we will do the country query first, and then the campus query second.