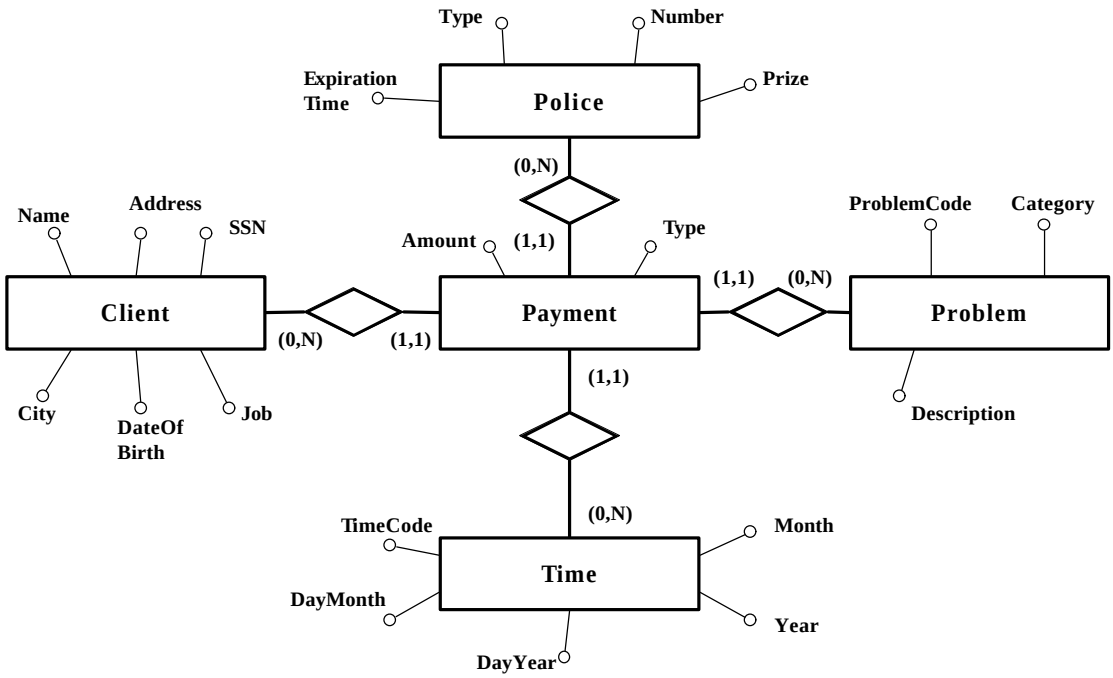
Chapter 13

Exercise 13.1

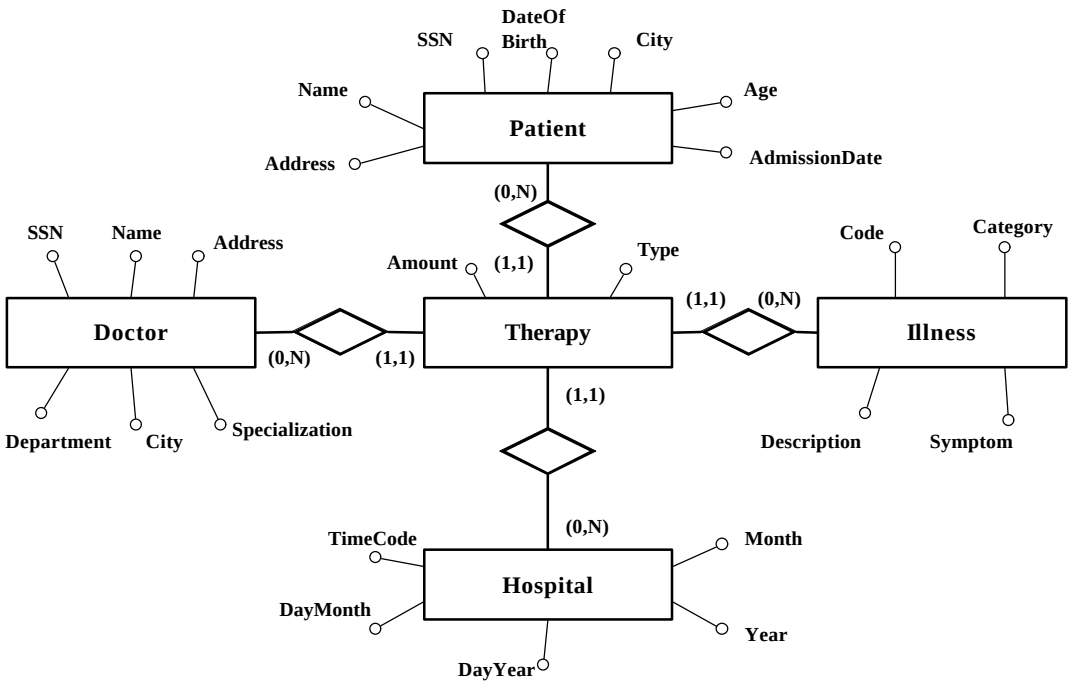
Complete the data mart projects illustrated in Figure 13.4 and Figure 13.5, identifying the attributes of fact and dimensions.

Sol:

1)



2)



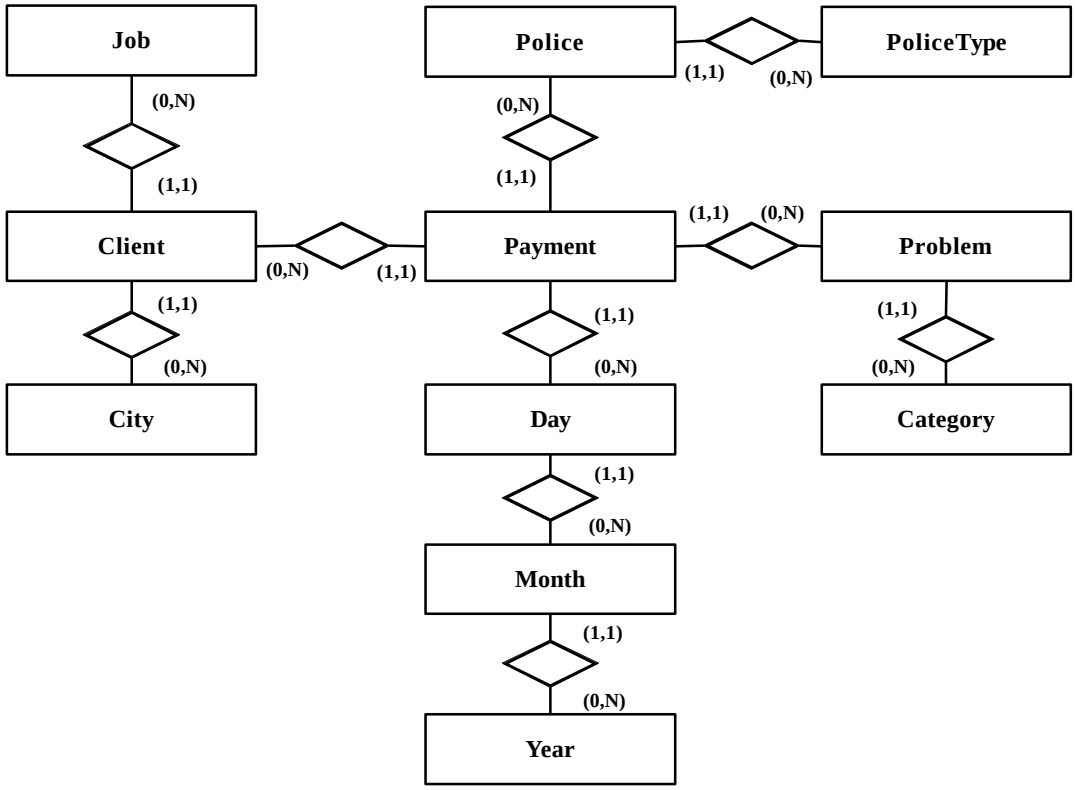
Exercise 13.2

Design the data marts illustrated in Figure 13.4 and Figure 13.5 identifying the hierarchies among the dimensions.

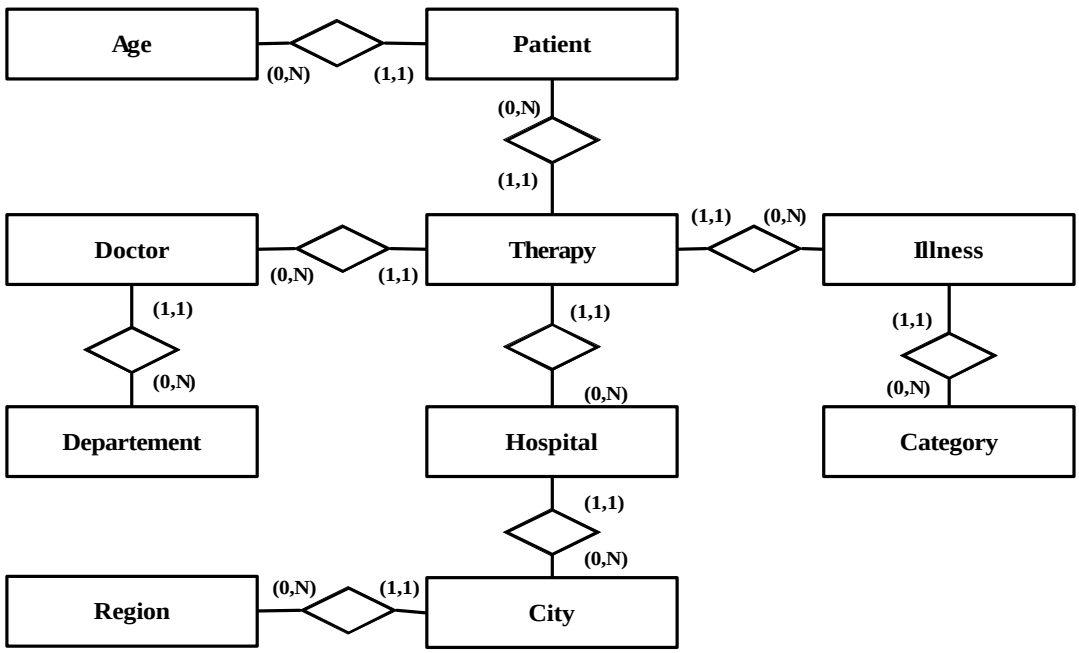
Sol:

SnowFlake schemas

1)



2)



Exercise 13.3

Refer to the data mart for the management of supermarkets, described in section 13.2.2. Design an interactive interface for the extraction of the data about classes of products sold in the various weeks of the year in stores located in large cities. Write the SQL query that corresponds to the proposed interface.

Sol:

We assume that attribute size refers to the City of the supermarket.

Product.Category	**Time.WeekYear**	**Market.Size**		**Qty**	*Schema*
Food Soap Saucepan	1..52	0...5000000			*Option*
		> 10000			*Condition*
Product.Category	Time.WeekYear			sum (Qty)	*View*

SQL code:

```
select Product.Category, Time.WeekYear, sum(Qty)
from Sale, Product, Time, Market
where Sale.ProductCode=product.ProductCode
  and Sale.MarketCode=Market.MarketCode
  and Sale.TimeCode=Time.TimeCode
  and Market.Size > 10000
group by Time.WeekYear, Product, Category
order by Product.Category, Time.WeekYear
```

Exercise 11.4

Describe the roll-up and drill-down operations relating to the result of the query posed in the preceding exercise.

Sol:

The following table contains a possible result for the query in Exercise 13.3

Product.Category	**Time.WeekYear**	**sum(Qty)**
Soap	15	20
Soap	30	30
Food	2	50
Food	6	80
Food	20	70
Saucepan	8	10
Saucepan	20	5

A drill-down operation adds a dimension to the analysis. In this example the operation add “City”. The new result of the query is:

Product.Category	**Time.WeekYear**	**Market.City**	**Sum (Qty)**
Soap	15	London	10
Soap	15	Edinburgh	10
Soap	30	London	20
Soap	30	Edinburgh	10
Food	2	London	30
Food	2	Edinburgh	20
Food	6	London	40
Food	6	Edinburgh	40
Food	20	London	70
Saucepan	8	London	5
Saucepan	8	Edinburgh	5
Saucepan	20	London	3
Saucepan	20	Edinburgh	2

A roll-up operation eliminates a dimension from the analysis: in this case the operation deletes the attribute WeekYear and re-aggregates the data:

Product.Category	**Market.City**	**sum(Qty)**
Soap	London	30
Soap	Edinburgh	20
Food	London	140
Food	Edinburgh	60
Saucepan	London	8
Saucepan	Edinburgh	7

Exercise 13.5

Describe the use of the **with cube** clause and **with roll up** clause in conjunction with the query posed in Exercise 13.3

Sol:

If the **with cube** clause is used in conjunction with a query, the result will contain all the possible aggregations of the dimensions of analysis:

```
select Product.Category, Time.WeekYear, sum(Qty)
from Sale, Product, Time, Market
where Sale.ProductCode=product.ProductCode
 and Sale.MarketCode=Market.MarketCode
 and Sale.TimeCode=Time.TimeCode
 and Market.Size > 10000
group by Time.WeekYear, Product, Category
with cube
```

The result is:

Product.Category	**Time.WeekYear**	**sum(Qty)**
Soap	15	20
Soap	30	30
Soap	ALL	50
Food	2	50
Food	6	80
Food	20	70
Food	ALL	200
Saucepan	8	10
Saucepan	20	5
Saucepan	ALL	15
ALL	2	50
ALL	6	80
ALL	8	10
ALL	15	20
ALL	20	75
ALL	30	30
ALL	ALL	265

The **roll up** clause causes a progressive aggregation of the dimensions: the aggregation is made from right to left, and so produces a smaller set of tuples than the **data cube:**

```
select Product.Category, Time.WeekYear, sum(Qty)
from Sale, Product, Time, Market
where Sale.ProductCode=product.ProductCode
 and Sale.MarketCode=Market.MarketCode
 and Sale.TimeCode=Time.TimeCode
 and Market.Size > 10000
group by Time.WeekYear, Product, Category
with roll up
```

Product.Category	**Time.WeekYear**	**sum(Qty)**
Soap	15	20
Soap	30	30
Food	2	50
Food	6	80
Food	20	70
Saucepan	8	10
Saucepan	20	5
Soap	ALL	50
Food	ALL	200
Saucepan	ALL	15
ALL	ALL	265

Exerxcise 13.6

Indicate a selection of bitmap indexes, join indexes and materialized views for the data mart described in Section 13.2.2.

Sol:

To indicate a set of indexes for a data warehouse it is necessary to know which are the most frequent operations applied to it.
The schema of the data warehouse is:

SALE(ProdCode, MarketCode, PromoCode, TimeCode, Qty, Revenue)
PRODUCT(ProdCode, Name, Category, SubCategory, Brand, Weight, Supplier)
MARKET(MarketCode, Name, City, Region, Zone, Size, Layout)
PROMOTION (PromoCode, Name, Type, Percentage, FlagCoupon, StartDate, EndDate, Cost, Agency)
TIME (TimeCode, DayWeek, DayMonth, DayYear, WeekMonth, WeekYear, MonthYear, Season, PreholidayFlag, HolidayFlag)

Let us suppose that the most frequent queries are:

1) Select all the sales with a specified product code and market code.
2) Select all the promotions with a specified start date and end date.
3) Select all the sales of a specified category of product.
4) Select all the sales with a specified promotion code and where category=”Food”.
5) Select all the sales in a specified Market where category=”Food”.

The first query suggests the introduction of two bitmap indexes on table SALE for the attributes ProductCode and MarketCode, because the query has a conjunction in its condition of selection.
Also, the second query suggests two bitmap indexes on table TIME, on the attributes StartDate and EndDate.

The third query requires a join between tables SALE and PRODUCT on the attribute ProdCode. A join index on this attribute makes the query more efficient.

The last two queries suggest the introduction of a view materialization for the query

```
select Sale.*, Product.Category
from Sale join Product on Sale.ProdCode=Product.prodCode
```

This query can be calculated only once and then can be used by both the queries 4 and 5 each time they need it.

This analysis does not consider the actual frequency of the queries, the frequency of update of the data and the time necessary to calculate the view. A more accurate analysis could show that the introduction of the materialized view is not always useful.

Exercise 13.7

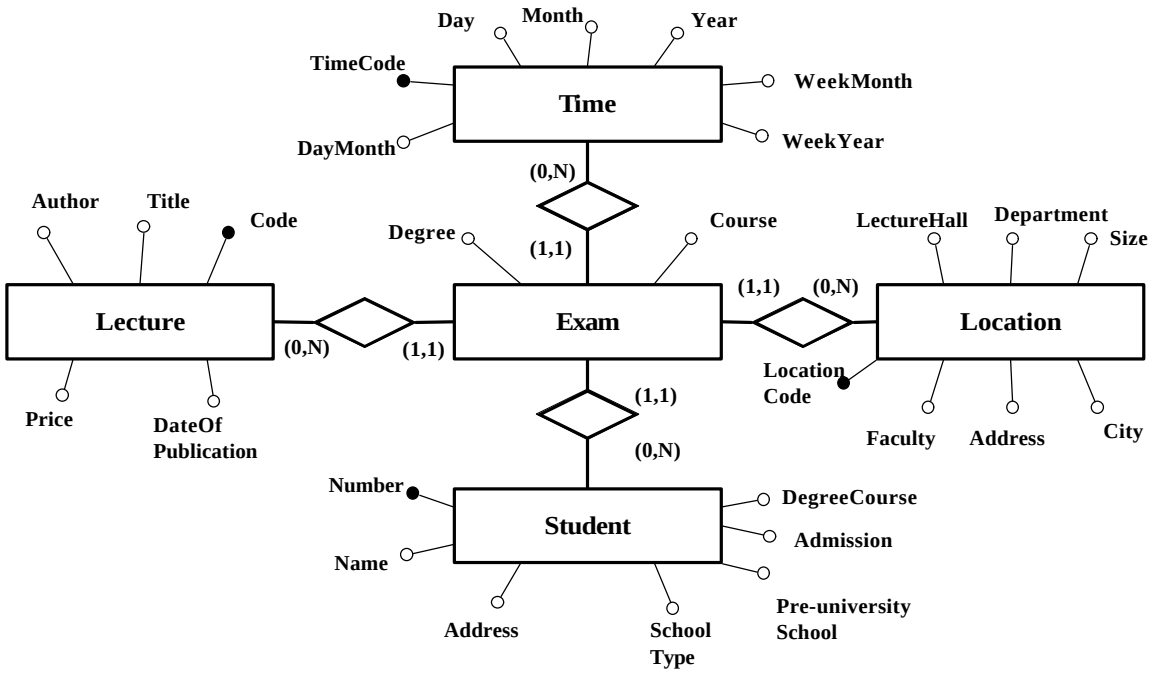
Design a data mart for the management of university exams. Use as facts the result of the exams taken by the students. Use as dimension the following:

1) time;
2) the location of the exam (supposing the faculty to be organized over more than one site);
3) the lecture involved;
4) the characteristics of the student (for example, the data concerning pre-university school records, grades achieved in the university admission exam, and chosen degree course).

Create both star schema and snowflake schema, and give their translation in relational form. Then express some interface for analysis simulating the execution of the **roll up** and **drill down** instructions. Finally, indicate a choice of bitmap indexes, join indexes and materialized views.

Sol:

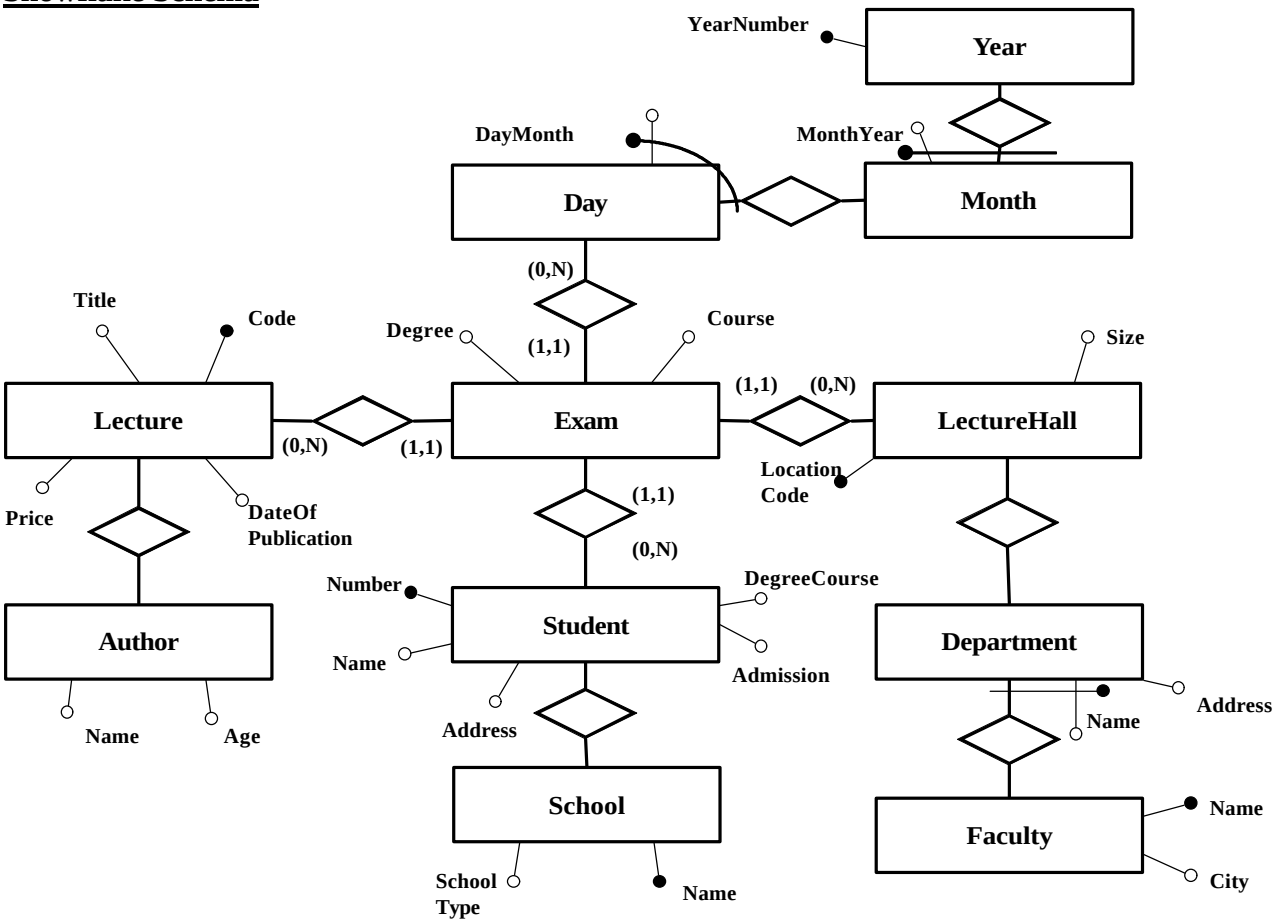
Star Schema



Relational Form

EXAM (<u>TimeCode</u>, <u>LocationCode</u>, <u>Student</u>, <u>LectureCode</u>, Course, Degree)
TIME(<u>TimeCode</u>, Day, Month, Year, WeekMonth, WeekYear, DayMonth)
LOCATION(<u>LocationCode</u>,LectureHall, Department, Faculty, Address, City)
STUDENT (<u>Number</u>, Name, Address, DegreeCourse, Admission, PreUniversitySchool, SchoolType)

Snowflake Schema



Relational Form

EXAM (DayMonth,MonthYear, YearNumber, LocationCode, Student, LectureCode, Course, Degree)
DAY(DayMonth,MontYear,YearNumber)
MONTH(MonthYear, YearNumber)
YEAR(YearNumber)
LECTURE (Code, Title, Author, Price, DateOfPublication)
AUTHOR (Name, Age)
STUDENT (Number, Name, Address, DegreeCourse, Admission)
SCHOOL (Name, SchoolType)
LECTUREHALL(LocationCode, Size, Department)
DEPARTMENT(Name, Faculty, Address)
FACULTY(Name, City)

Interface for analysis:

Lecture.Author	**Location.Faculty**		*****	*Schema*
[string]	Engineering Math Physics			*Option*
Smith, Brown, Green	Engineering, Math			*Condition*
	Location.Faculty		count(*)	*View*

This interface selects the number of exams taken by students in the Faculty of Engineering and Math, who read books of Smith, Brown and Green to prepare the exams.

Result of the query:

Lecture.Author	**Location.Faculty**	**count(*)**
Smith	Engineering	270
Brown	Engineering	264
Brown	Math	250
Green	Math	280

Drill down on Time.MonthYear attribute:

Lecture.Author	**Location.Faculty**	**Time.MonthYear**	**count(*)**
Smith	Engineering	2	70
Smith	Engineering	6	100
Smith	Engineering	9	100
Brown	Engineering	2	264
Brown	Math	3	160
Brown	Math	7	90
Green	Math	1	150
Green	Math	7	130

Roll up on Lecture.Author

Location.Faculty	**Time.MonthYear**	**count(*)**
Engineering	2	334
Engineering	6	100
Engineering	9	100
Math	1	150
Math	3	160
Math	7	220

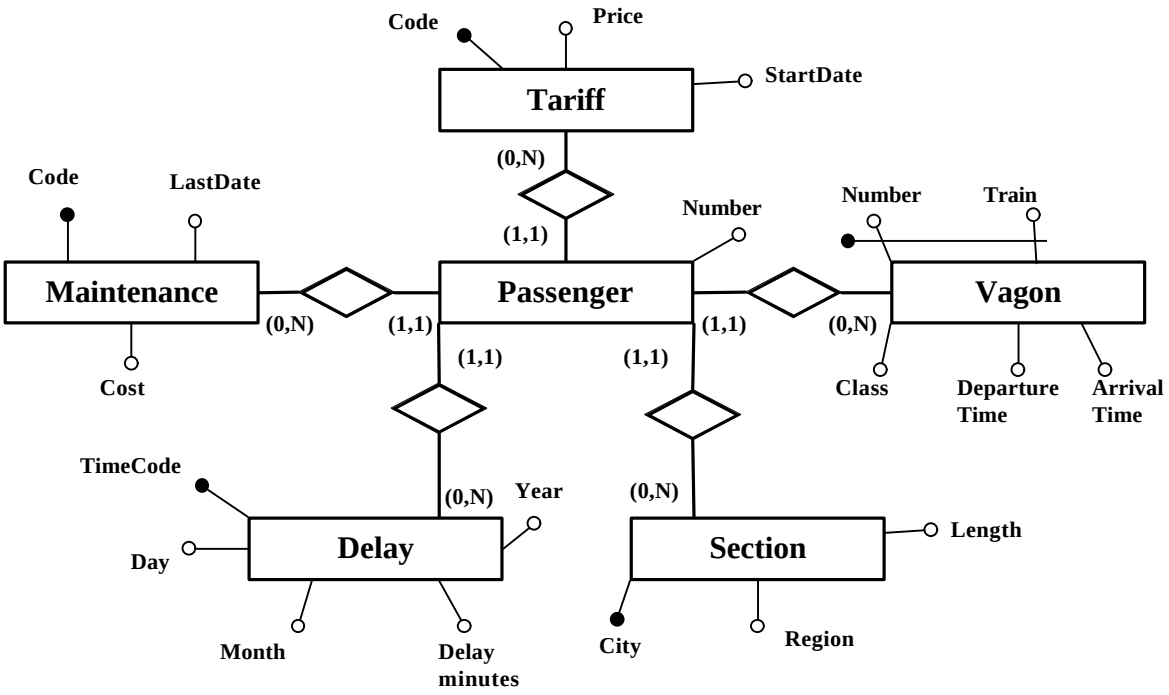
These queries suggest the introduction of bitmap indexes on Location.Faculty and Lecture.Author.
To make the joins more efficient it is possible to introduce join indexes (on Location.Code, Lecture.Code and Time.TimeCode) or materialized views.
The choice depends on the dimension of the tables, the number of different values of the attributes and the update frequency of the tables
Considering that in table Exam the tuples are often added but rarely updated or deleted, the choice of materialized views may be the best.

Exercise 13.8

Design one or more data marts for railway management: use as facts the total number of daily passengers for each tariff on each train and on each section of the network. As dimensions, use the tariffs, the geographical position of the cities on the network, the composition of the train, the network maintenance and the daily delays.
Create one or more star schemas and give their translation in relational form.

Sol:

Star Schema:



Relational Schema

PASSENGER(Tariff, Maintenance, Delay, Section, Vagon, Train, Number)
TARIFF (Code, Price, StartDate)
MAINTENANCE(Code, LastDate, Cost)
DELAY(TimeCode, Day, Month, Year, DelayMinutes)
SECTION(City, Region, Length)
VAGON(Number, Train, Class, DepartureTime, ArrivalTime)

Exercise 13.9

Consider the database in Figure 13.19. Extract the association rules with support and confidence higher or equal to 20 percent. Then indicate which rules are extracted if a support higher than 50 percent is requested.

Transaction	Date	Goods	Qty	Price
1	17/12/98	ski-pants	1	140
1	17/12/98	boots	1	180
2	18/12/98	ski-pole	1	20
2	18/12/98	T-shirt	1	25
2	18/12/98	jacket	1	200
2	18/12/98	boots	1	70
3	18/12/98	jacket	1	200
4	19/12/98	jacket	1	200
4	19/12/98	T-shirt	3	25
5	20/12/98	T-shirt	1	25
5	20/12/98	jacket	1	200
5	20/12/98	tie	1	25

Figure 13.19

Sol:

Premise	Consequence	Support	Confidence
ski-pant	boots	0.2	1
boots	ski-pants	0.2	0.5
T-shirt	jacket	0.6	1
jacket	T-shirt	0.6	0.75
T-shirt	ski-pole	0.2	0.33
ski-pole	T-shirt	0.2	1
T-shirt	boots	0.2	0.33
boots	T-shirt	0.2	0.5
jacket	boots	0.2	0.25
boots	jacket	0.2	0.5
jacket	tie	0.2	0.25
tie	jacket	0.2	1
T-shirt	tie	0.2	0.33
tie	T-shirt	0.2	1
{ski-pole, T-shirt}	{jacket, boots}	0.2	1
{jacket, boots}	{ski-pole, T-shirt}	0.2	1
{ski-pole, jacket}	{T-shirt, boots}	0.2	1
{T-shirt, boots}	{ski-pole, jacket}	0.2	1
{ski-pole, boots}	{T-shirt,jacket}	0.2	1
{T-shirt,jacket}	{ski-pole, boots}	0.2	0.33
{boots,ski-pole, T-shirt}	jacket	0.2	1
jacket	{boots,ski-pole, T-shirt}	0.2	0.25
{ski-pole, T-shirt,jacket}	boots	0.2	1
boots	{ski-pole, T-shirt,jacket}	0.2	0.5
{ski-pole,jacket,boots}	T-shirt	0.2	1
T-shirt	{ski-pole,jacket,boots}	0.2	0.33

{jacket,boots,T-shirt}	ski-pole	0.2	1
ski-pole	{jacket,boots,T-shirt}	0.2	1
{T-shirt,jacket}	tie	0.2	0.33
tie	{T-shirt,jacket}	0.2	1
{T-shirt,tie}	jacket	0.2	1
jacket	{T-shirt,tie}	0.2	0.25
{jacket,tie}	T-shirt	0.2	1
T-shirt	{jacket,tie}	0.2	0.33

If a support higher than 50 percent is requested, the only rule is:

- T-shirt -> jacket

Exercise 13.10

Discretize the prices of the database in Exercise 13.9 into three values (low, average and high). Transform the data so that for each transaction a single tuple indicates the presence of at least one sale for each class. Then construct the association rules that indicates the simultaneous presence in the same transaction of sales belonging to different price classes.
Finally, interpret the results.

Sol:

Discretization of prices:

low: price ≤ 25
average: 25 < prize ≤ 200
high prize ≥ 200

Transaction	**Date**	**Qty**	**Class**
1	17/12/98	2	Average
2	18/12/98	1	Average
2	18/12/98	2	Low
2	18/12/98	1	High
3	18/12/98	1	High
4	19/12/98	1	High
4	19/12/98	3	Low
5	20/12/98	2	Low
5	20/12/98	1	High

Association rules

Premise	**Consequence**	**Support**	**Confidence**
Average	High	0.4	0.5
High	Average	0.4	0.25
Average	Low	0.2	0.5
Low	Average	0.2	0.33
High	Low	0.6	0.75
Low	High	0.6	1
{Low, Average}	High	0.2	1
High	{Low, Average}	0.2	0.25
{High, Low}	Average	0.2	0.33
Average	{High, Low}	0.2	1
{High, Average}	Low	0.2	1
Low	{High, Average}	0.2	0.33

These results show that the most important rules are:

- Low -> High
- High -> Low

The biggest quantity of sales refers to the low class.
It means that high and low class articles are often bought together, while average class articles are not very pleasant for buyers.
These results may be useful in locating the articles in the various sectors of a supermarket.

Exercise 13.11

Describe a database for car sales with the description of the automobiles (sport cars, saloons, estate, etc), the cost and the cylinder capacity of the automobiles (discretized in classes), and the age and salary of the buyers (also discretized into classes). Then form hypotheses on the structure of a classifier showing the propensity of the purchase of cars by different categories of persons.

Sol:

The database is composed of the following tables:

CAR (Number, Model, Color, Optional, Cost)
MODEL(Code, Name, CylinderCap, MaxSpeed, Category)
CLIENT(Name, Age, Salary)

Example of an instance of database:

CAR

Number	Model	Color	Options	Cost
1	2478	red	air conditioned	High
2	2478	black		High
3	2631	white	radio	Average
4	4932	red		Low

MODEL

Model	Name	Cylinder	Category
2478	Ferrari	3000-4000	SportCar
2631	BMW	2000-3000	StationWagon
4932	Toyota	1000-2000	Runabout

CLIENT

Name	Age	Salary
Green	20-25	Low
Brown	25-30	High
Smith	30-40	Average
Thomson	40-50	Average

Classifier:

Propensity to purchase a car.