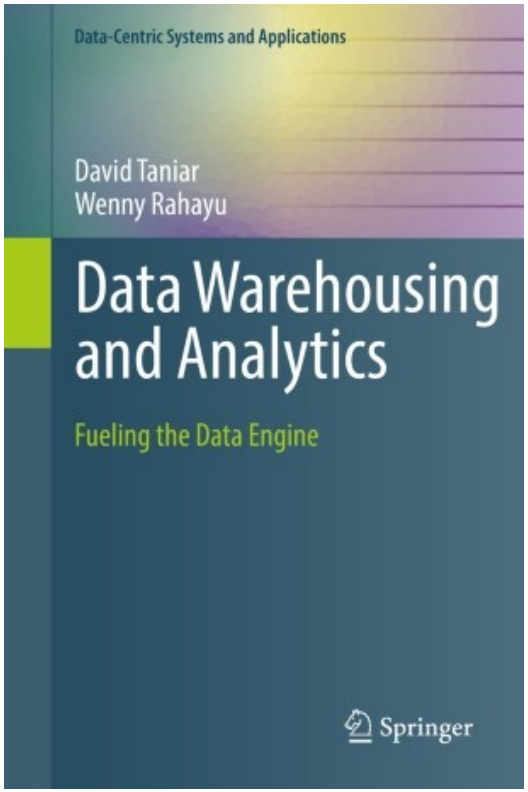


Chapter 3
Creating Facts and Dimensions:
More Complex Processes

Overview

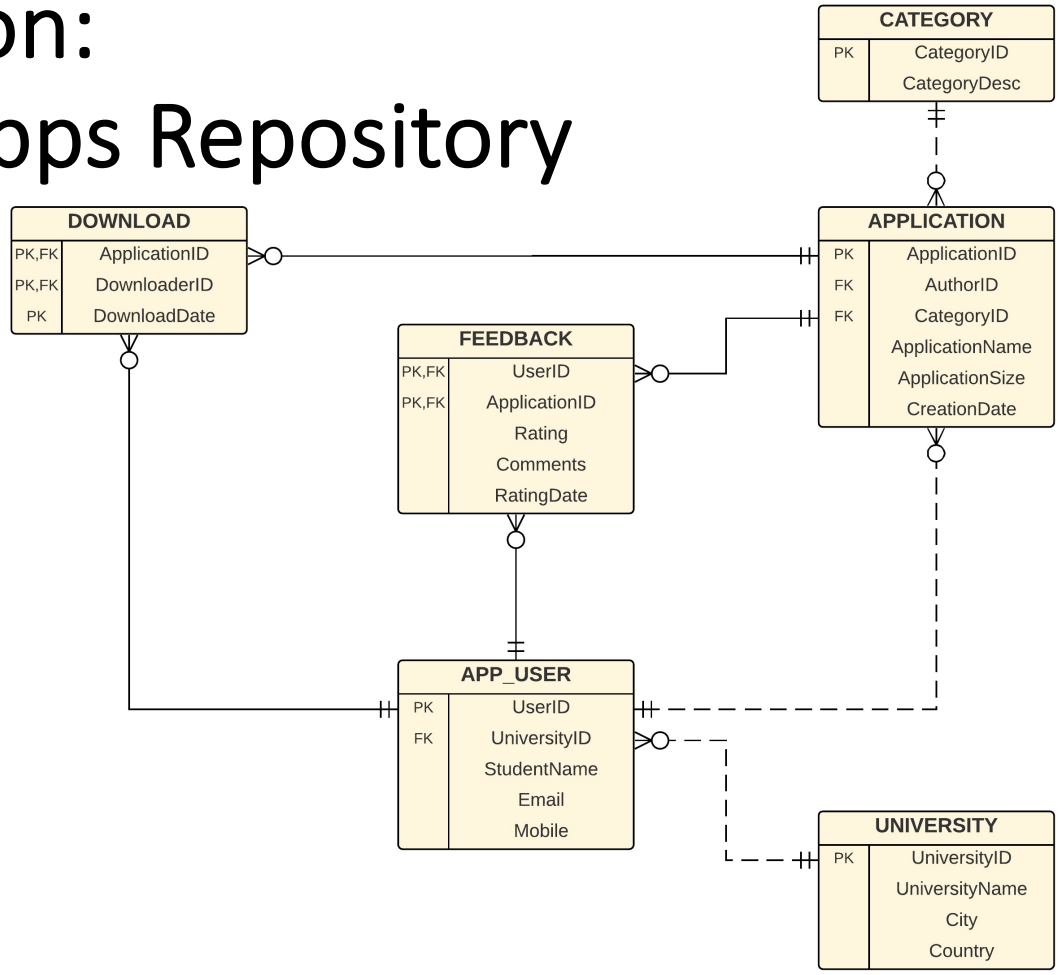
1. Use of `count` Function
2. Average in the Fact
3. Outer Join
4. Creating Temporary Dimension Tables
5. Creating Temporary Tables in the Operational Database

1. Use of Count Function

- SQL:
  - `count(*)` → number of records
  - `count(attribute)` → exclude null values
  - `count(distinct attribute)` → remove duplication

1. Use of Count Function: Example: Mobile apps Repository

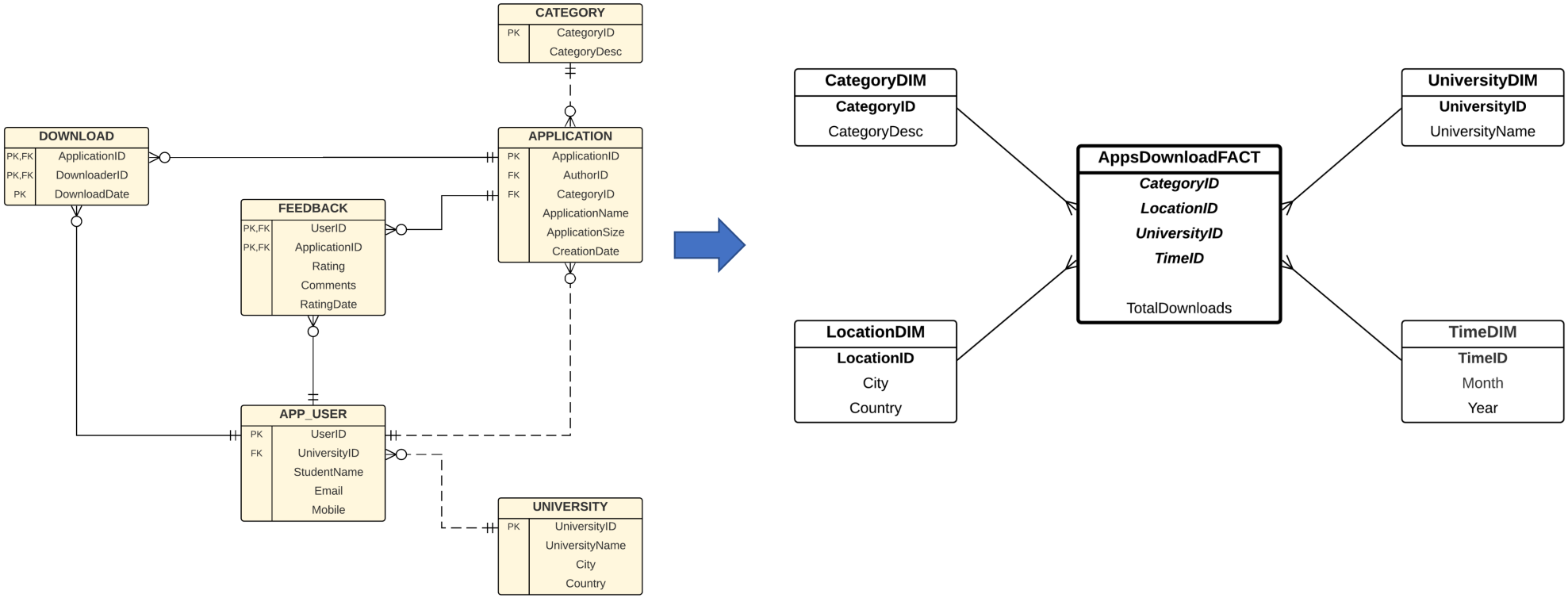
- Monalisa App Store that allows students from any university in the world to publish their applications and receive feedbacks



1. Use of Count Function:
Example: Mobile apps Repository

- Star Schema is needed to analyse:
  - Rating, Feedbacks
  - By different authors, apps

1. Use of Count Function:
Example: Mobile apps Repository



1. Use of Count Function:
Example: Mobile apps Repository

- Dimensions: Category, University, Location, and Time

```
create table CategoryDim as
select * from Category;

create table UniversityDim as
select UniversityID, UniversityName from University;

create table LocationDim as
select distinct
  Country || City as LocationID,
  City,
  Country
from University;

create table TimeDim as
select distinct
  to_char(DownloadDate, 'YYYYMM') as TimeID,
  to_char(DownloadDate, 'MM') as Month,
  to_char(DownloadDate, 'YYYY') as Year
from Download;
```

1. Use of Count Function:
Example: Mobile apps Repository

- Create a TempFact

```
create table TempFact as
select
  to_char(D.DownloadDate, 'YYYYMM') as DownloadMonth,
  to_char(A.CreationDate, 'YYYYMM') as CreationMonth,
  U.Country || U.City as LocationID,
  A.CategoryID,
  A.ApplicationID,
  U.UniversityID
from University U, App_User R, Download D, Application A
where
  U.UniversityID = R.UniversityID and
  R.UserID = D.DownloaderID and
  D.ApplicationID = A.ApplicationID;
```

1. Use of Count Function:
Example: Mobile apps Repository

- Create the Fact (Total Downloads)

```
create table AppsDownloadFact as
select
  DownloadMonth as TimeID,
  LocationID,
  CategoryID,
  UniversityID,
  count(*) as TotalDownloads
from TempFact
group by
  DownloadMonth,
  LocationID,
  CategoryID,
  UniversityID;
```

1. Use of Count Function:
Example: Mobile apps Repository

- Suppose the Fact is Total Apps, instead of Total Downloads

```
create table AppsFact as
select
  CreationMonth as TimeID,
  LocationID,
  CategoryID,
  UniversityID,
  count(distinct ApplicationID) as TotalApps
from TempFact
group by
  CreationMonth,
  LocationID,
  CategoryID,
  UniversityID;
```

2. Average in the Fact

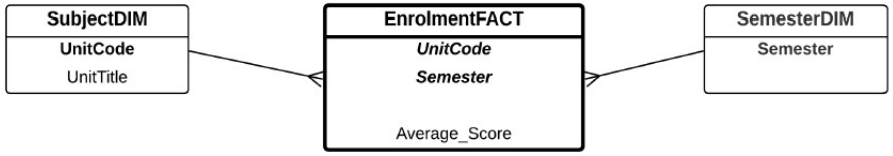
- Many aggregate functions can be used in fact table
- There are some pitfalls, especially when fact measures are highly aggregated
- Inappropriate use of average function will be highlighted

2. Average in the Fact:

Average of an Average Example

Unit Code	Unit Title	Semester	Student First Name	Score
IT001	Database	1	Mirriam	81
IT001	Database	1	Allan	41
IT001	Database	1	Ben	74
IT001	Database	1	Kate	85
IT001	Database	1	Larry	87
IT001	Database	1	Leonard	75
IT001	Database	2	Juan	64
IT001	Database	2	Andy	32
IT002	Java	1	Ally	65
IT002	Java	1	Menson	47
IT002	Java	2	Mirriam	78
IT002	Java	2	Ben	73
IT002	Java	2	Larry	64
IT003	SAP	1	Ally	63
IT004	Network	2	Juan	53
IT004	Network	2	Menson	52

- Fact: Average Score
- Dimension: Subject, Semester



2. Average in the Fact:
Average of an Average Example

Table 1.2 Table: Fact

Unit Code	Semester	Average Score
IT001	1	73.833
IT001	2	48
IT002	1	56
IT002	2	71.667
IT003	1	63
IT004	2	52.5

Table 1.3 Table: Subject Dimension

Unit Code	Unit Title
IT001	Database
IT002	Java
IT003	SAP
IT004	Network

Table 1.4 Table: Semester Dimension

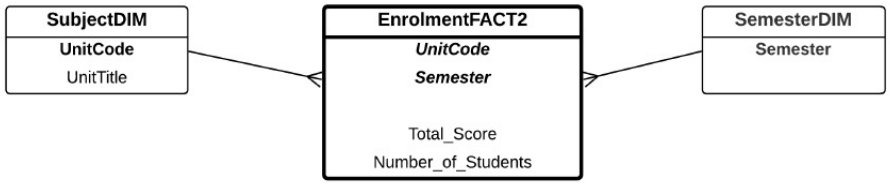
Semester
1
2

- Average Score for the Database Unit from Fact table (73.833+48)/2= **60.9165**
- Average Score for the Database unit from operational table 539/8=**67.375**
- Storing average as a fact measurement is not a good idea
- Replace **Average Score** with **Total Score** and **Number of Students**

2. Average in the Fact:
Average of an Average Correction

Table: Fact Version 2

Unit Code	Semester	Total Score	Number of Students
IT001	1	443	6
IT001	2	96	2
IT002	1	112	2
IT002	2	215	3
IT003	1	63	1
IT004	2	105	2




```
select
  sum(Total_Score)/sum(Number_of_Students)
  as Average_Score
from EnrolmentFact2
where UnitCode  = 'IT001';
```

2. Average in the Fact: *Min Max* Example

- Min and Max will always have global value
- Min and Max can be used in Fact table
- Mixing between Min and Max will be meaningless
- Count and Sum are commonly found in fact table

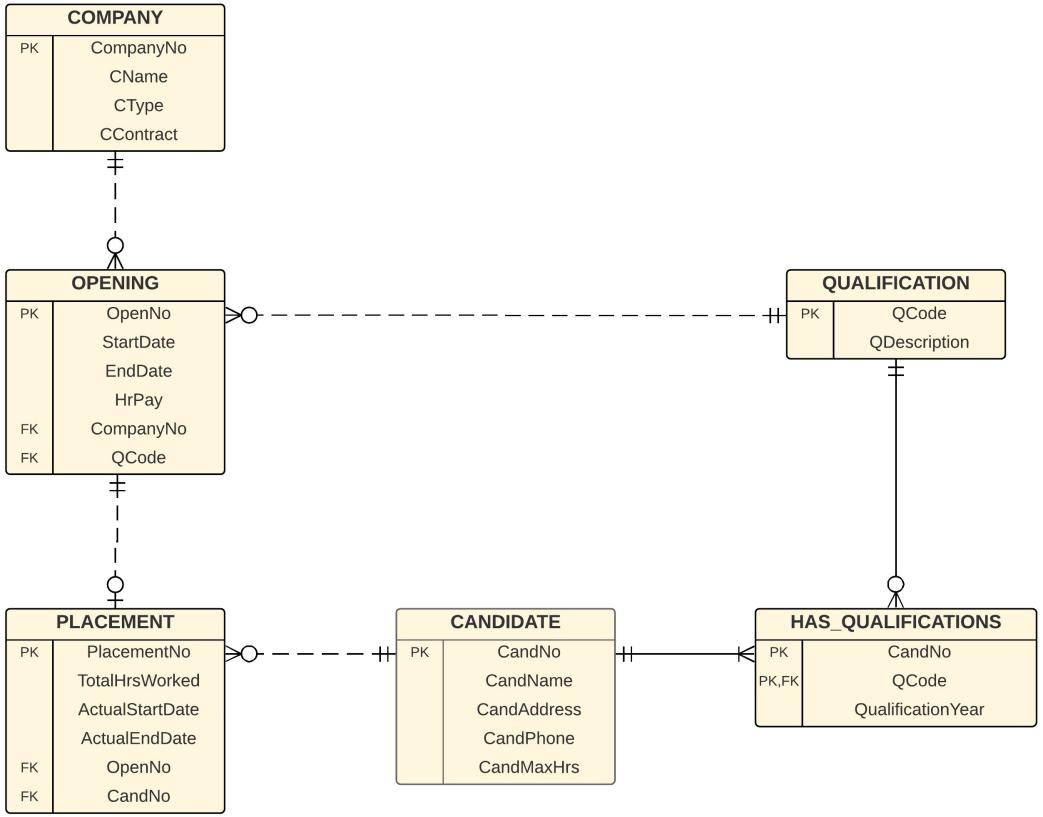
Table: Fact Version 3

Unit Code	Semester	Min Score	Max Score
IT001	1	41	87
IT001	2	32	64
IT002	1	47	65
IT002	2	64	78
IT003	1	63	63
IT004	2	52	53

```
select max(Max_Score)
from EnrolmentFact3
where UnitCode  = 'IT001';

select min(Min_Score)
from EnrolmentFact3
where UnitCode  = 'IT001';
```

3. Outer Join (Employment Agency)

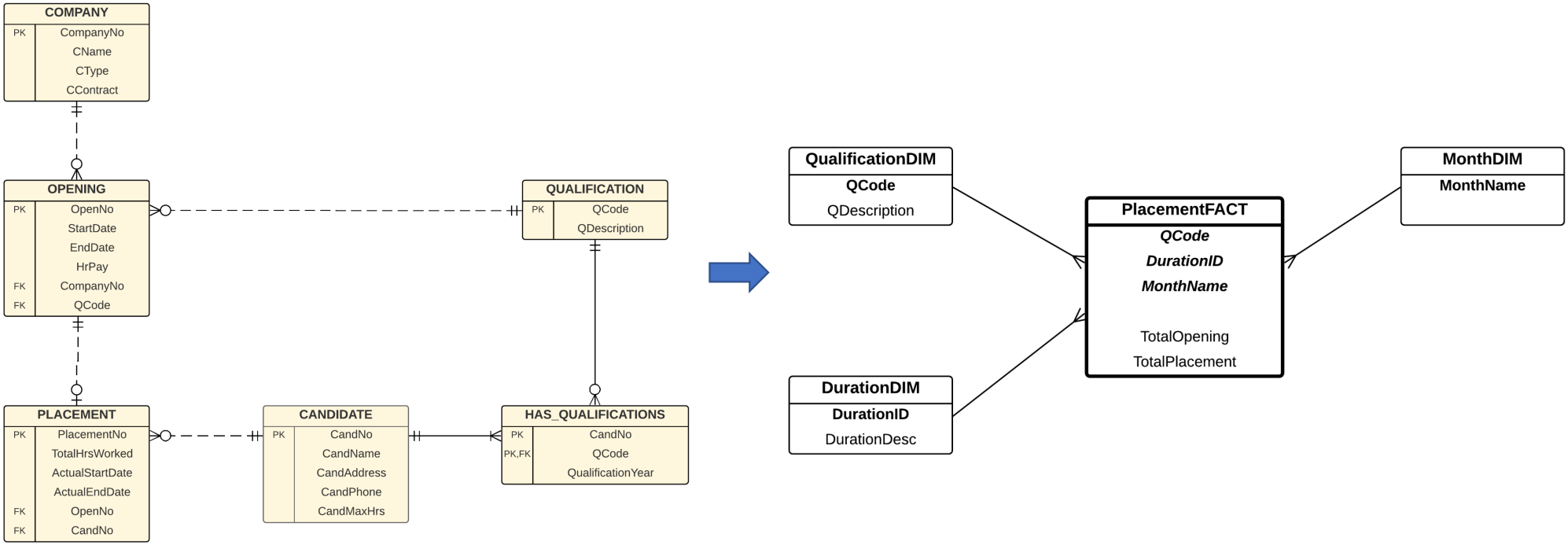
- Employment Agency which places temporary workers in companies during peak periods
- Business process is described in the chapter



3. Outer Join
Example: Employment Agency

- Star schema is needed for analysis
- Fact measures:
  - Total Opening
  - Total Placement
- Dimension:
  - Duration
  - Qualification
  - Month

3. Outer Join
Example: Employment Agency



3. Outer Join
Example: Employment Agency

- Create dimension tables

```
create table QualificationDim as
select * from Qualification;

create table MonthDim as
select
  distinct to_char(ActualStartDate, 'Month')
  as MonthName
from Placement;

create table DurationDim
(DurationID number,
 DurationDesc varchar2(20));

insert into DurationDim values (1, 'Short-Term');
insert into DurationDim values (2, 'Medium-Term');
insert into DurationDim values (3, 'Long-Term');
```

3. Outer Join
Example: Employment Agency

- Duration Dimension table is created manually → need TempFact

```
create table TempFact as
select
  O.QCode,
  O.StartDate,
  O.EndDate,
  to_char(P.ActualStartDate, 'Month') as MonthName,
  O.OpenNo,
  P.CandNo
from Opening O left outer join Placement P
  on O.OpenNo = P.OpenNo;
```

```
alter table TempFact
add (DurationID number);

update TempFact
set DurationID = 1
where EndDate - StartDate < 10;

update TempFact
set DurationID = 2
where EndDate - StartDate >= 10
and EndDate - StartDate <=30;

update TempFact
Set DurationID = 3
where EndDate - StartDate > 30;
```

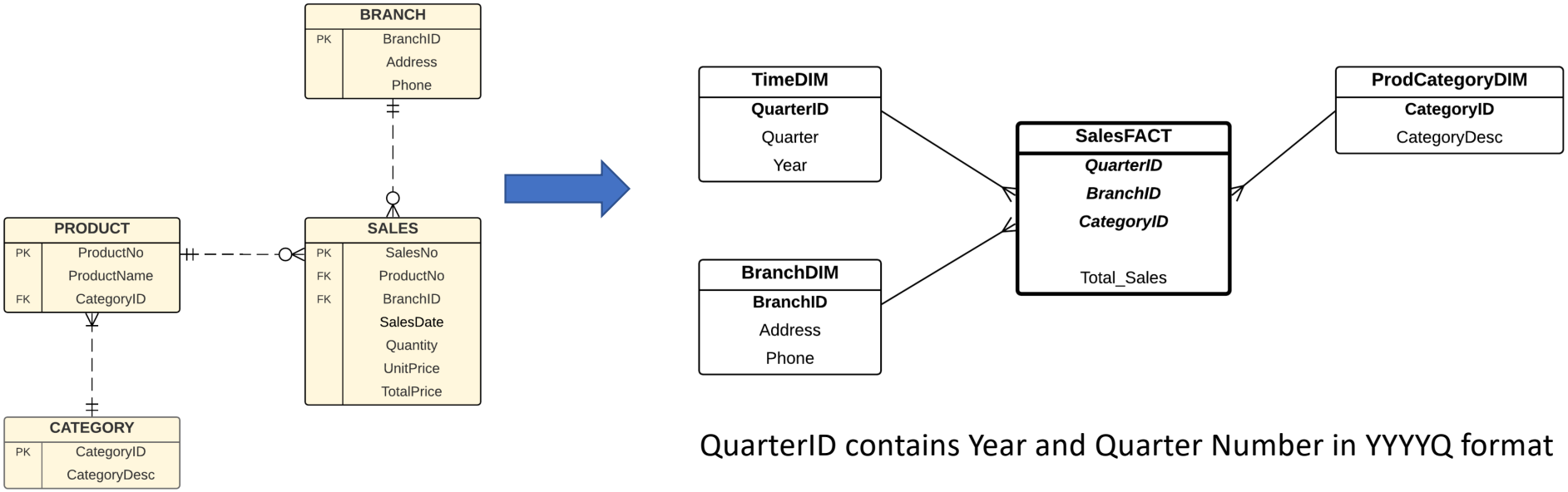
3. Outer Join
Example: Employment Agency

- Create the Fact

```
create table AgencyFact as
select
  QCode, DurationID, MonthName,
  count(OpenNo) as TotalOpening,
  count(CandNo) as TotalPlacement
from TempFact
group by QCode, DurationID, MonthName;
```

4. Creating *Temporary* Dimension Tables
Sales Example

- Required dimension table cannot be created directly.



QuarterID contains Year and Quarter Number in YYYYQ format

4. Creating *Temporary* Dimension Tables Sales Example

```
create table BranchDim as
select * from Branch;

create table ProdCategoryDim as
select * from Category;
```

- The Time Dimension that has QuarterID, Quarter, and Year attributes need to be created manually.
- Problems:
  - Number of records is unknown
  - Manual insertion in not efficient
- Temporary Time dimension is needed

4. Creating *Temporary* Dimension Tables Sales Example

- Create a Temporary Time Dimension:

```
create table TimeDimTemp as
select distinct
  to_char(SalesDate, 'MM') as Month,
  to_char(SalesDate, 'YYYY') as Year
from Sales;

alter table TimeDimTemp add
(QuarterID char(5),
 Quarter   char(1));

update TimeDimTemp
set Quarter = '1'
where Month >= '01'
and Month <= '03';

update TimeDimTemp
set Quarter = '2'
where Month >= '04'
and Month <= '06';

update TimeDimTemp
set Quarter = '3'
where Month >= '07'
and Month <= '09';

update TimeDimTemp
set Quarter = '4'
where Month >= '10'
and Month <= '12';

update TimeDimTemp
set QuarterID = Year||Quarter;

create table TimeDim as
select distinct QuarterID, Quarter, Year
from TimeDimTemp;
```

4. Creating *Temporary* Dimension Tables Sales Example

- The rest of the step

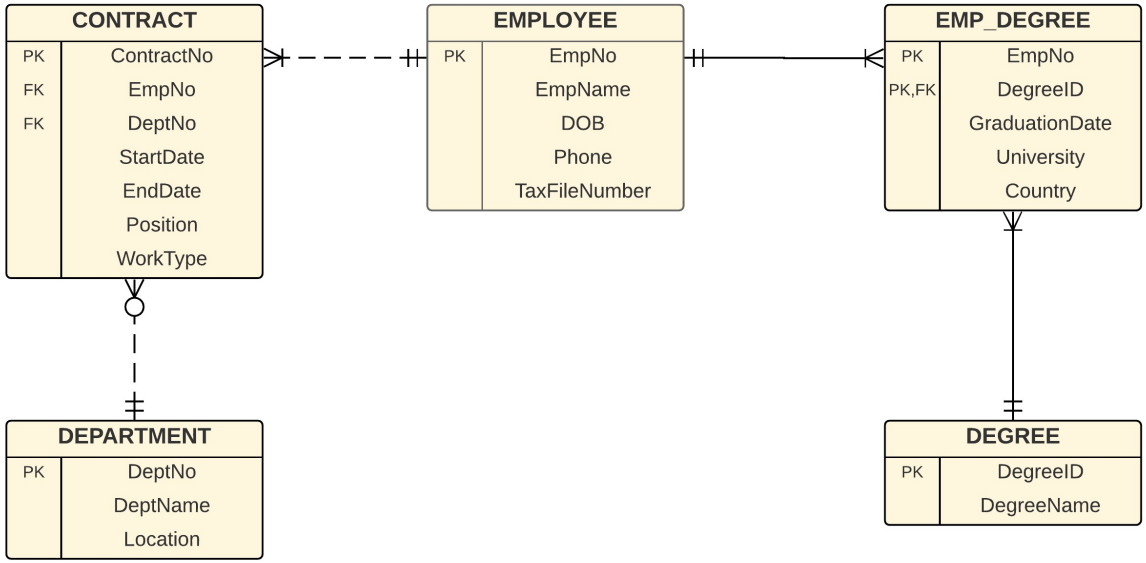
```
create table TempFact as
select
  to_char(S.SalesDate, 'YYYY') as Year,
  to_char(S.SalesDate, 'MM') as Month,
  B.BranchID,
  P.CategoryID,
  S.TotalPrice
from Branch B, Sales S, Product P
where B.BranchID = S.BranchID
and S.ProductNo = P.ProductNo;

alter table TempFact
add (Quarter char(1));

update TempFact
set Quarter = '1'
where Month >= '01'
and Month <= '03';

update TempFact
set Quarter = '2'
where Month >= '04'
and Month <= '06';

update TempFact
set Quarter = '3'
where Month >= '07'
and Month <= '08';

update TempFact
set Quarter = '4'
where Quarter is null;

alter table TempFact
add (QuarterID char(5));

update TempFact
set QuarterID = Year||Quarter;

create table SalesFact as
select
  QuarterID,
  BranchID,
  CategoryID,
  sum(TotalPrice) as Total_Sales
from TempFact
group by QuarterID, BranchID, CategoryID;
```

5. Creating *Temporary* Tables in the Operational Database

- The Tables from the operational database might need to go through further transformation before they are ready to be used to create the Fact Table.
- A case study will be used to show how an operational database table is further transformed before being used to create the data warehouse.
- This case study is about hiring sessional jobs in the university.

5. Creating *Temporary* Tables in the Operational Database

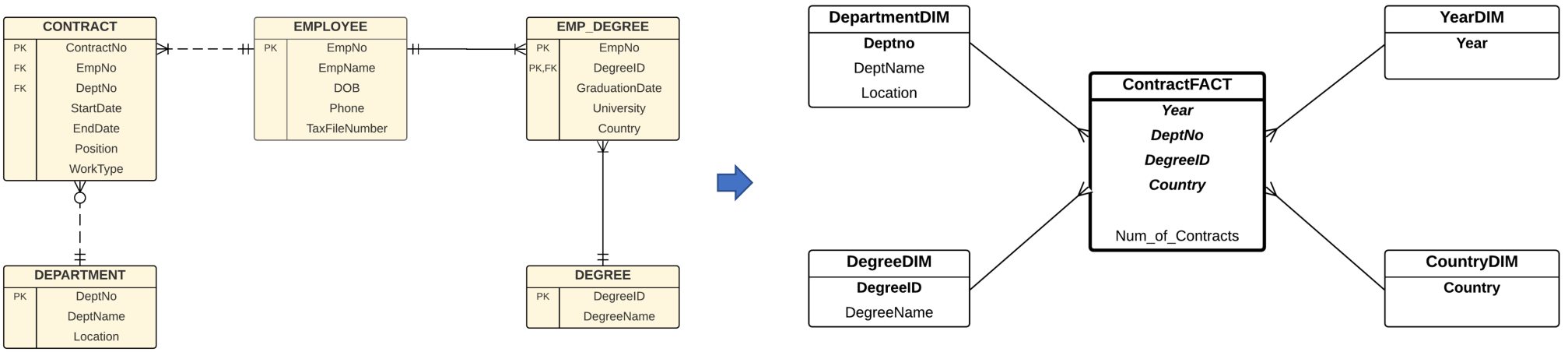
- University employs its students to do various jobs, such as tutoring, programming, administration, etc. These jobs are called "Sessional" jobs.
- Business process is described in the chapter



5. Creating *Temporary* Tables in the Operational Database

- Star schema is needed for analysis.
- Fact:
  - Number of contract
- Dimension:
  - Department
  - Year
  - Country
  - Degree

5. Creating *Temporary* Tables in the Operational Database



5. Creating *Temporary* Tables in the Operational Database

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

create table DegreeDim as
select * from Degree;

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

- The first three dimensions can be created directly.
- An Employee may have several degrees, but the data warehouse needs on the latest degree
- Employee table needs transformation to get the latest degree for each employee → EmployeeTemp

5. Creating *Temporary* Tables in the Operational Database

```
create table EmployeeTemp as
select
  T.EmpNo, T.EmpName, T.DOB,
  T.Phone, T.TaxFileNumber, T.DegreeID
from (
  select
    E.EmpNo, E.EmpName, E.DOB, E.Phone,
    E.TaxFileNumber, D.DegreeID,
    rank() over
      (partition by E.EmpNo
      order by D.GraduationDate desc) as Rank
  from Employee E, Emp_Degree D
  where E.EmpNo = D.EmpNo) T
where T.Rank = 1;
```

```
create table ContractFact as
select
  E.DegreeID,
  to_char(C.StartDate, 'YYYY') as Year,
  C.DeptNo,
  count(*) as Num_of_Contracts
from EmployeeTemp E, Contract C
where E.EmpNo = C.EmpNo
group by
  E.DegreeID,
  to_char(C.StartDate, 'YYYY'),
  C.DeptNo;
```

Summary

- Complex processing in creating Fact Table
  - Aggregate function
  - Outer Join operation
- Complex processing in creating dimension tables
  - Temporary dimensions
- Pre-processing on operational data → Temporary operational database tables