

Advanced Data Warehousing

Multi Operational Databases as the Input to a Data Warehouse

1. **The Philharmonic Society** is a Monash student university orchestra. They rehearse once a week, and have public concerts a few times a year. Each member plays a musical instrument. They also have a professional conductor from outside Monash. When they perform concerts, they sell tickets. Their operational database contains information about their members, concert activities, and Expenditure.

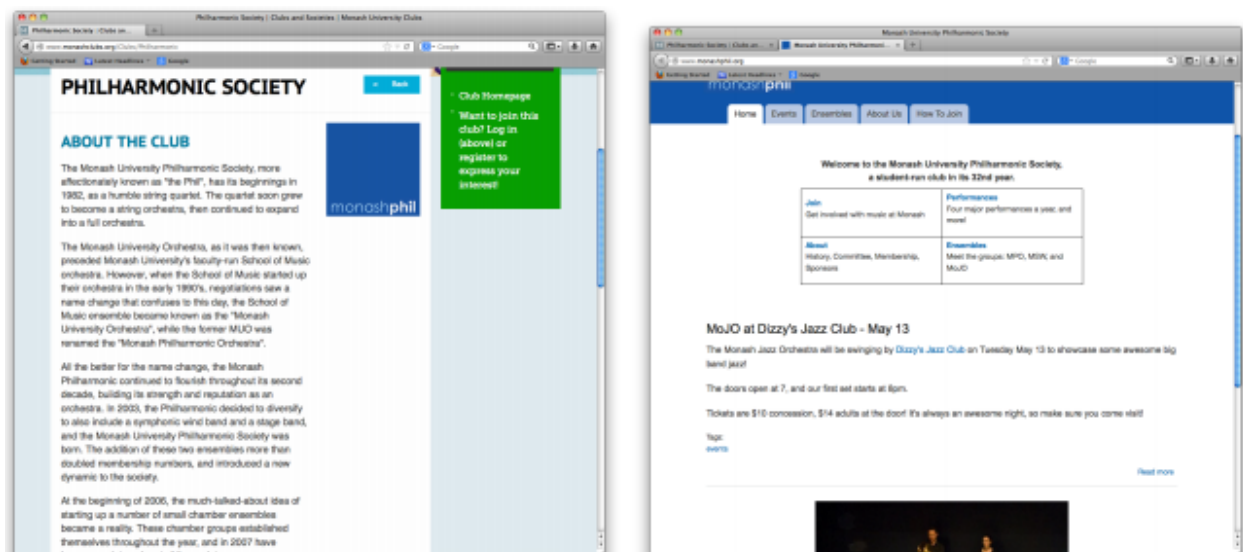


Figure 1 - Philharmonic Society

The club main event is Concert. The concert's **Hosting Cost** (Booking Revenue, Advertising, Ticket Printing, etc.) and the **Conductor Fee** are paid by the club **Expenditure**. The club's **Income** is from concerts' **Ticket Sales**, there are 2 types of tickets: Normal and Concession. The Philharmonic Society has no membership fees.

Total Expenditure = Total (Hosting Cost + Conductor Fee)

Total Income = Total Ticket Sales

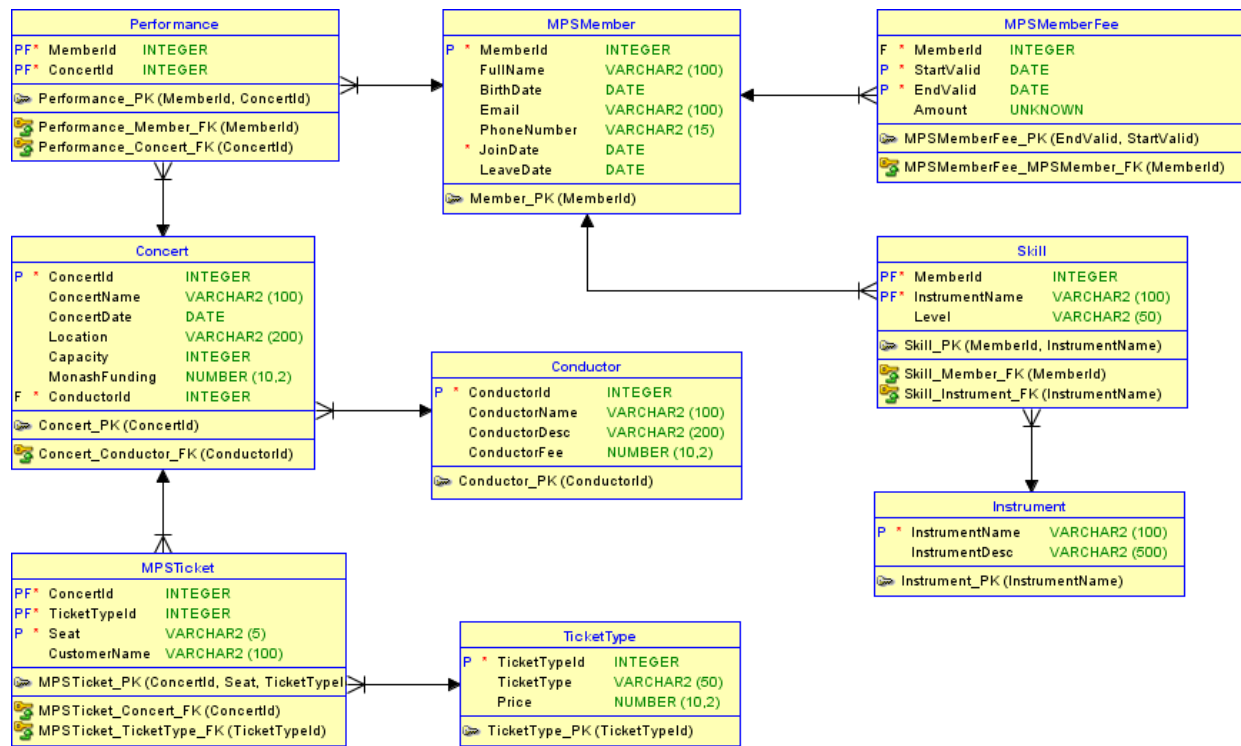


Figure 2 - Philharmonic Society Database

2. The Business and Commerce Students' Society hosts events relating to job orientation, seminars, etc. They have some companies as their sponsors. Their operational database contains information about their members, events and seminars, as well as expenditure, which may include financial supports from sponsors.

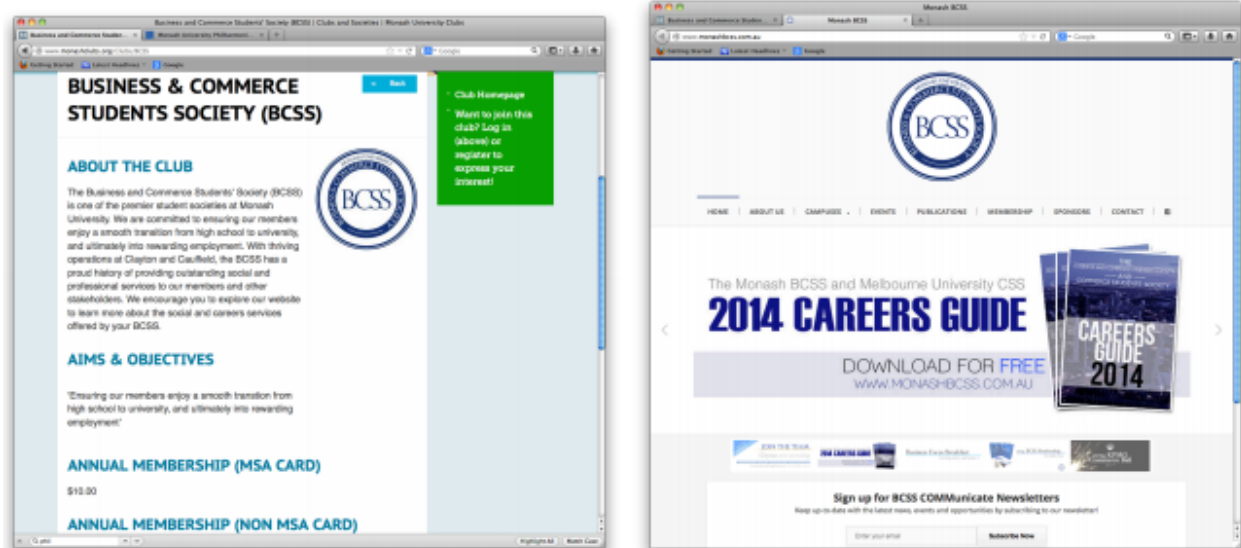


Figure 3 - Business and Commerce Students' Society

This club has membership fee. Each membership card is valid for 1 year and needs to be renewed every year. The club holds 3 types of events: **Professional Events**, **General Events**, and **Social Events** - <http://www.monashbcss.com.au/events/>. A **member** will get a fixed **discount** while a **non-member** has to pay a **full price** for the **event's ticket**. The club **Expenditure** is sum of annual funding from sponsors (including Monash) and membership fees. This **Expenditure is allocated to all events**. Sponsors advertise many programs via the club site (e.g. internship programs) - <http://www.monashbcss.com.au/sponsors/>. Monash University would like to know the **activity Expenditure** and the **activity Income** of the club. Other Incomes from membership fee and annual funding are not taken into account.

Total Expenditure = Total (Allocated Expenditure)

Total Income = Total (Ticket Price * Number of Tickets + Ticket Price * (1 – Discount) * Number of Member Registration)

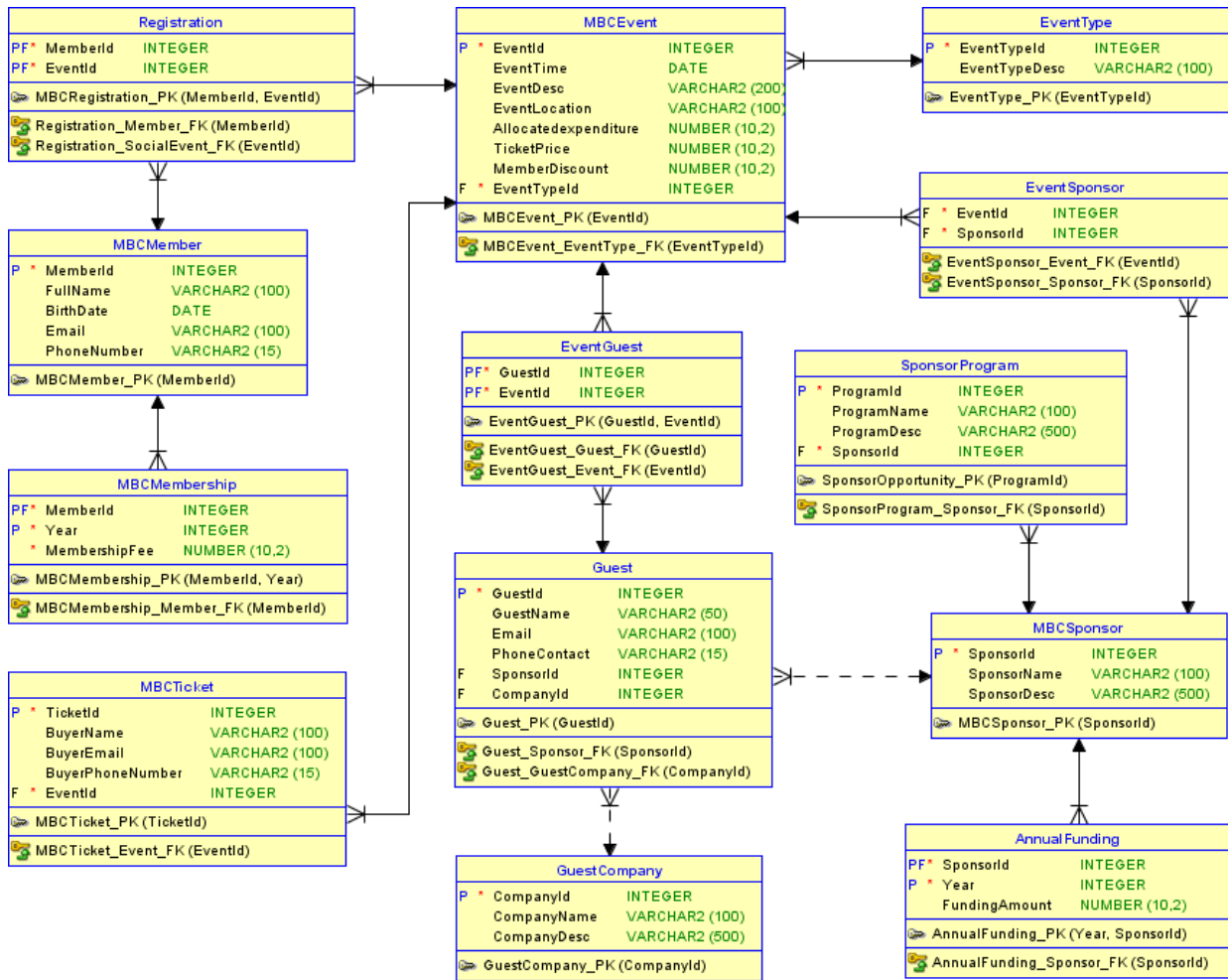


Figure 4 - Business and Commerce Students' Society Database

3. The Japanese Club is a cultural club, which contains members who are interested in the Japanese culture. This may include students who study Japanese at Monash or any students who can speak Japanese and would like to intermingle with Japanese native speakers or other students who speak Japanese. Their events are more cultural events, as well as some social events. Their operational database contains information about their members, events, and Expenditure.

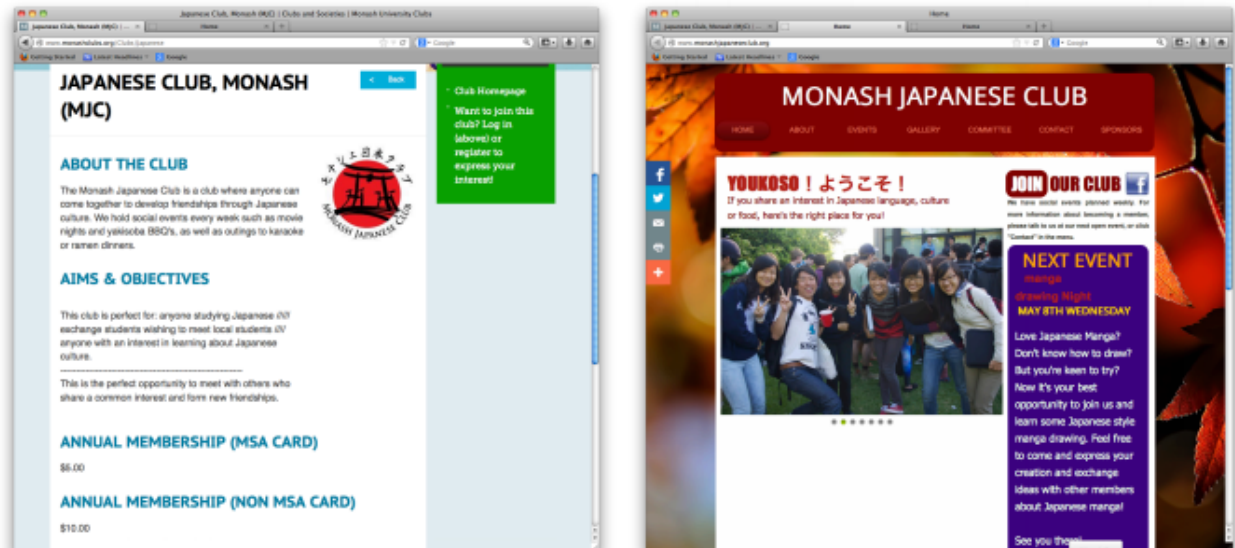


Figure 5 - Japanese Club

This club does not require membership fee. It has only one type of event: **Culture/Social Event**. The club gets **funding for each event**, but does **not receive annual funding** from sponsors. Monash is also one of their sponsors.

Total Expenditure = Total (Event Funding Amount)

Total Income = Total (Member Ticket Count * Member Price + Non-Member Ticket Count * Non-Member Price)

This club maintains data in an excel file:

	A	B	C	D	E	F
1	EVENTID	EVENTTIME	EVENTDESC	EVENTLOCATION	MEMBERPRICE	NONMEMBERPRICE
2	1	01-FEB-14	Okonomiyaki BBQ	Back bar,67 green st, windsor, vic	40	45
3	2	05-MAR-14	Trivia Night	Imperial 522 Chapel St South Yarra VIC 3141	30	32
4	3	06-APR-13	Onigiri Night	The Vic 281 Victoria St Abbotsford VIC 3067	15	20
5	4	09-MAY-13	Origami Night	Little Red Pocket Cocktail Bar 422 Little Collins St Melbourne VIC 3000	32	36
6	5	11-JUN-12	Karaoke	Melbourne Karaoke Lounge 465 Spencer St West Melbourne VIC 3003	26	32
7						

EVENTFUNDING MJCSPONSOR **SOCIALEVENT** MEMBER1 ...

Figure 6 - The Monash Japanese Club Excel DB

The equivalent ER Diagram (SQL Developer can import/export Database from/to Excel Files):

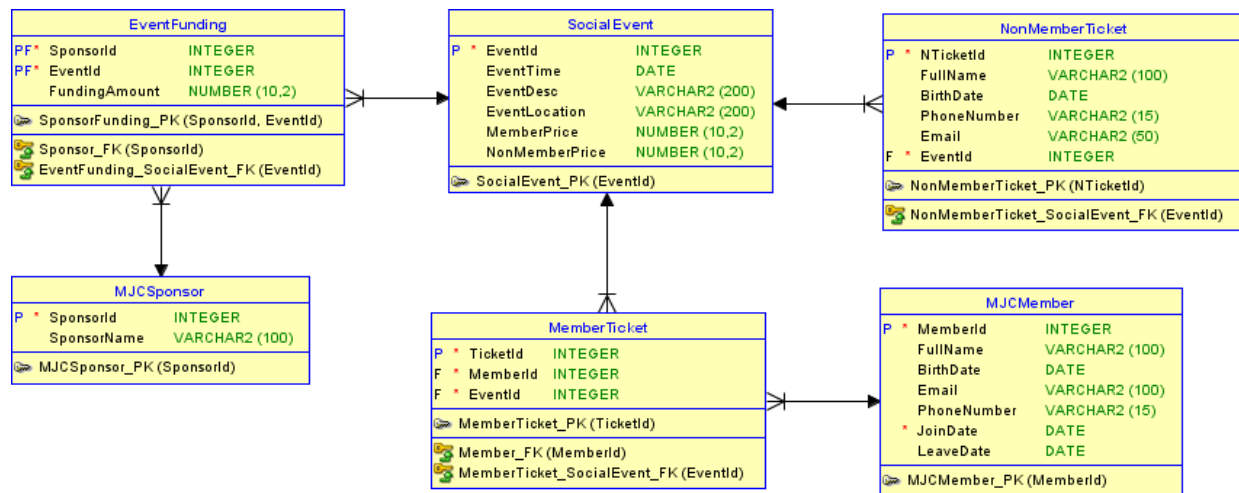


Figure 7 - Japanese Club Database

The following diagram shows that there are three input operational databases which create one data warehouse.

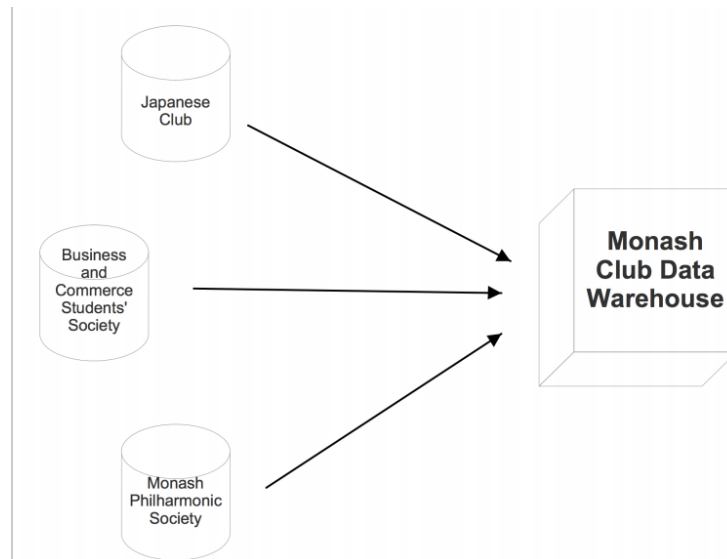


Figure 8 - Multi Operational Database

The information stored in each of these operational databases are members, activities, and financial Expenditure. Although they are different operational databases, they have the same theme. So, it becomes possible to create one data warehouse which combines information from these different operational databases.

The following diagram shows a star schema which combines the three operational databases. For simplicity, the star schema has only two dimensions: club and year. The star schema focuses on total activity Expenditure and total activity Income.

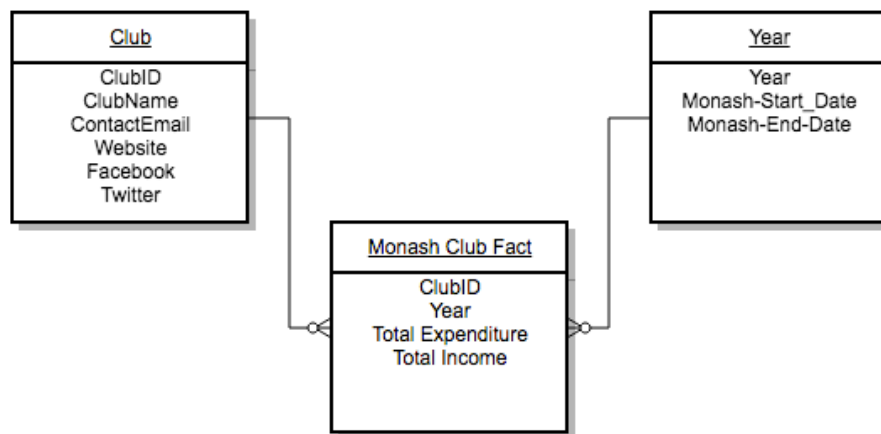


Figure 9 - Monash Club Star Schema

```
-- MONASH CLUB STAR SCHEMA
-- Create Club Dimension
CREATE TABLE ClubDim (
  ClubId          INTEGER,
  ClubName        VARCHAR(100),
  ContactEmail    VARCHAR(100),
  Website         VARCHAR(100),
  Facebook        VARCHAR(100),
  Twitter         VARCHAR(100)
);

INSERT INTO ClubDim
VALUES(1, 'Monash University Philharmonic Society',
'gro.sbulchsanom@cinomrahlihp', 'http://www.monashphil.org',
'http://www.facebook.com/monashphil', null);

INSERT INTO ClubDim
VALUES(2, 'The Business and Commerce Students' Society (BCSS)',
'gro.sbulchsanom@sscb', 'http://www.monashbcss.com.au',
'https://www.facebook.com/monashBCSS', null);

INSERT INTO ClubDim
VALUES(3, 'Monash Japanese Club', 'gro.sbulchsanom@esenapaj',
'http://www.monashjapaneseclub.org/',
'https://www.facebook.com/groups/5723718069/', null);

COMMIT;
```

CLUBID	CLUBNAME	CONTACTEMAIL	WEBSITE	FACEBOOK	TWITTER
1	Monash University Philharmonic...	gro.sbulchsan...	http://www...	http://www...	(null)
2	The Business and Commerce Stud...	gro.sbulchsan...	http://www...	https://www...	(null)
3	Monash Japanese Club	gro.sbulchsan...	http://www...	https://www...	(null)

Figure 10 – Club Dimension Table


```
-- Create Year Dimension
CREATE TABLE MBCYearDim
AS SELECT DISTINCT TO_CHAR(EventTime, 'yyyy') AS Year
FROM MBCEvent;

CREATE TABLE MJCYearDim
AS SELECT DISTINCT TO_CHAR(EventTime, 'yyyy') AS Year
FROM SocialEvent;

CREATE TABLE MPSYearDim
AS SELECT DISTINCT TO_CHAR(ConcertDate, 'yyyy') AS Year
FROM Concert;

CREATE TABLE YearDim AS
SELECT * FROM MJCYearDim UNION
SELECT * FROM MPSYearDim UNION
SELECT * FROM MBCYearDim;

ALTER TABLE YearDim ADD (
    Monash_Start_Date DATE,
    Monash_End_Date    DATE
);

UPDATE YearDim
SET Monash_Start_Date = TO_DATE('01-Mar-'||Year, 'dd-mm-yyyy');

UPDATE YearDim
SET Monash_End_Date = TO_DATE('01-Dec-'||Year, 'dd-mm-yyyy');

COMMIT;
```




	 YEAR	 MONASHSTARTDATE	 MONASHENDDATE
1	2012	01-MAR-12	01-DEC-12
2	2013	01-MAR-13	01-DEC-13
3	2014	01-MAR-14	01-DEC-14

Figure 11 – Year Dimension Table

```
-- The Philharmonic Society
-- MPS Temp Fact
CREATE TABLE MPSTempFact1 AS
SELECT
    T.ConcertId,
    TO_CHAR(CT.ConcertDate, 'yyyy') AS Year,
    CT.HostingCost,
    CR.ConductorFee,
    TT.Price
FROM
    MPTicket T, TicketType TT, Concert CT, Conductor CR
WHERE
    T.TicketTypeId = TT.TicketTypeId and
    CT.ConcertId = T.ConcertId and
    CT.ConductorId = CR.ConductorId;
```

CONCERTID	YEAR	HOSTINGCOST	CONDUCTORFEE	PRICE
1	2012	1000	2000	150
1	2012	1000	2000	150
1	2012	1000	2000	150
1	2012	1000	2000	150
1	2012	1000	2000	150
1	2012	1000	2000	150
1	2012	1000	2000	150
1	2012	1000	2000	150
1	2012	1000	2000	150
1	2012	1000	2000	150

Figure 12 – MPS TempFact1 Table

```
-- Table Concert has a higher level of aggregation than MPTicket so we need
-- to use "group by" when calculate the Income
CREATE TABLE MPSTempFact2 AS
SELECT
    ConcertId,
    Year,
    HostingCost,
    ConductorFee,
    SUM(Price) AS Income
FROM
    MPSTempFact1
group by ConcertId, Year, HostingCost, ConductorFee;
order by ConcertId, Year, HostingCost, ConductorFee;
```

CONCERTID	YEAR	HOSTINGCOST	CONDUCTORFEE	INCOME
1	2012	1000	2000	5950
2	2013	4000	1000	6250
3	2014	3000	1000	6200
4	2014	2000	2000	6050
5	2013	5000	2000	6200

Figure 13 – MPS TempFact2 Table (Without ClubID)

```

ALTER TABLE MPSTempFact2
ADD (ClubId INTEGER);

UPDATE MPSTempFact2
SET ClubId = 1;

```

CONCERTID	YEAR	HOSTINGCOST	CONDUCTORFEE	INCOME	CLUBID
1	2012	1000	2000	5950	1
2	2013	4000	1000	6250	1
3	2014	3000	1000	6200	1
4	2014	2000	2000	6050	1
5	2013	5000	2000	6200	1

Figure 14 – MPS TempFact2 Table (With ClubID)

```

-- MPS Fact
CREATE TABLE MPSFact AS
SELECT ClubId, Year,
       SUM(HostingCost + ConductorFee) AS TotalExpenditure,
       SUM(Income) AS TotalIncome
FROM MPSTempFact2
GROUP BY ClubId, Year;

```

CLUBID	YEAR	TOTALEXPENDITURE	TOTALINCOME
1	2012	3000	5950
1	2014	8000	12250
1	2013	12000	12450

Figure 15 – MPS Fact Table

```
--Monash BCSS
--Calculate number of registrations (member tickets) for each event
create table EventRegistration as
select E.EventId, count(*) as RegistrationCount
from MBCEvent E, Registration R
where E.EventId = R.EventId
group by E.EventId;
```

EVENTID	REGISTRATIONCOUNT
1	10
2	10
4	9
5	8
3	10

Figure 16 – Number of Registration For Each Event

```
--Calculate number of normal tickets for each event
create table EventTicket as
select E.EventId, count(*) as TicketCount
from MBCEvent E, MBCTicket T
where E.EVENTID = T.EVENTID
group by E.EventId;
```

EVENTID	TICKETCOUNT
1	6
2	6
4	6
5	6
3	6

Figure 17 – Number of Normal Tickets For Each Event

```
--MBCTempFact:
--drop table MBCTempFact cascade constraints purge;
create table MBCTempFact as
select E.EventId, TO_CHAR(E.EventTime, 'yyyy') AS Year,
       E.TICKETPRICE, E.ALLOCATEDEXPENDITURE, E.MEMBERDISCOUNT,
       T.TICKETCOUNT, R.REGISTRATIONCOUNT
from MBCEvent E, EventTicket T, EventRegistration R
where  E.EventId = T.EventId
and    E.EventId = R.EventId;
```

EVENTID	YEAR	TICKETPRICE	ALLOCATEDEXPENDITURE	MEMBERDISCOUNT	TICKETCOUNT	REGISTRATIONCOUNT
1	2014	200	1000	0.5	6	10
2	2013	150	1200	0.2	6	10
4	2012	200	1200	0.1	6	9
5	2012	150	2200	0.2	6	8
3	2013	150	1800	0.1	6	10

Figure 18 – MBC Temp Fact Table(Without ClubID)

```
ALTER TABLE MBCTempFact
ADD (ClubId INTEGER);

UPDATE MBCTempFact
SET CLUBID = 2;
```

EVENTID	YEAR	TICKETPRICE	ALLOCATEDEXPENDITURE	MEMBERDISCOUNT	TICKETCOUNT	REGISTRATIONCOUNT	CLUBID
1	2014	200	1000	0.5	6	10	2
2	2013	150	1200	0.2	6	10	2
4	2012	200	1200	0.1	6	9	2
5	2012	150	2200	0.2	6	8	2
3	2013	150	1800	0.1	6	10	2

Figure 19 – MBC Temp Fact Table(With ClubID)

```
--MBC Fact
CREATE TABLE MBCFact AS
SELECT ClubId, Year,
       SUM(ALLOCATEDEXPENDITURE) AS TotalExpenditure,
       SUM(RegistrationCount * (1-MemberDiscount) * TicketPrice + TicketCount *
       TicketPrice) AS TotalIncome
FROM MBCTempFact
GROUP BY ClubId, Year;
```

CLUBID	YEAR	TOTALEXPENDITURE	TOTALINCOME
2	2014	1000	2200
2	2013	3000	4350
2	2012	3400	4680

Figure 20 – MBC Fact Table

```
-- Monash Japanese Club
-- Calculate number of member tickets for each event
create table EventMemberTicket as
select E.EventId, count(*) as MemberTicketCount
from SocialEvent E, MemberTicket M
where E.EventId = M.EventId
group by E.EventId;
```

EVENTID	MEMBERTICKETCOUNT
1	5
2	4
4	4
5	4
3	4

Figure 21 – Number of Member Tickets For Each Event

```
-- Calculate number of non-member tickets for each event
create table EventNonMemberTicket as
select E.EventId, count(*) as NonMemberTicketCount
from SocialEvent E, NonMemberTicket M
where E.EventId = M.EventId
group by E.EventId;
```

EVENTID	NONMEMBERTICKETCOUNT
1	5
2	5
4	5
5	5
3	5

Figure 22 – Number of Non-Member Tickets For Each Event

```
-- MJC TempFact
CREATE TABLE MJCTempFact AS
SELECT
    E.EventId,
    TO_CHAR(E.EventTime, 'yyyy') AS YEAR,
    F.FundingAmount,
    M.MemberTicketCount,
    N.NonMemberTicketCount,
    E.MemberPrice,
    E.NonMemberPrice
FROM
    SocialEvent E,
    EventNonMemberTicket N,
    EventMemberTicket M,
    EventFunding F
WHERE E.EventId = N.EventId
and E.EventId = M.EventId
and E.EventId = F.EventId;
```

EVENTID	YEAR	FUNDINGAMOUNT	MEMBERTICKETCOUNT	NONMEMBERTICKETCOUNT	MEMBERPRICE	NONMEMBERPRICE
1	2014	400	5	5	80	100
1	2014	450	5	5	80	100
2	2014	400	4	5	40	60
2	2014	300	4	5	40	60
3	2013	500	4	5	20	40
4	2013	300	4	5	40	60
4	2013	420	4	5	40	60
5	2012	500	4	5	60	80
3	2013	200	4	5	20	40
1	2014	600	5	5	80	100

Figure 23 – MJC Temp Fact Table (Without ClubID)

```
ALTER TABLE MJCTempFact
ADD (ClubId INTEGER);
```

```
UPDATE MJCTempFact
SET ClubId = 3;
```

EVENTID	YEAR	FUNDINGAMOUNT	MEMBERTICKETCOUNT	NONMEMBERTICKETCOUNT	MEMBERPRICE	NONMEMBERPRICE	CLUBID
1	2014	400	5	5	80	100	3
1	2014	450	5	5	80	100	3
2	2014	400	4	5	40	60	3
2	2014	300	4	5	40	60	3
3	2013	500	4	5	20	40	3
4	2013	300	4	5	40	60	3
4	2013	420	4	5	40	60	3
5	2012	500	4	5	60	80	3
3	2013	200	4	5	20	40	3
1	2014	600	5	5	80	100	3

Figure 24 – MJC Temp Fact Table (With ClubID)

-- MJC Fact

```
CREATE TABLE MJCFact AS
SELECT ClubId, Year,
       SUM(FundingAmount) AS TotalExpenditure,
       SUM(MemberPrice * MemberTicketCount +
           NonMemberPrice * NonMemberTicketCount) AS TotalIncome
FROM MJCTempFact
GROUP BY ClubId, Year;
```

CLUBID	YEAR	TOTALEXPENDITURE	TOTALINCOME
3	2014	2150	3620
3	2013	1420	1480
3	2012	500	640

Figure 20 – MJC Fact Table

-- Monash Club Fact

```

CREATE TABLE MonashClubFact AS
SELECT * FROM MPSFact UNION
SELECT * FROM MJCFact UNION
SELECT * FROM MBCFact;

SELECT * FROM MonashClubFact;

COMMIT;

```

CLUBID	YEAR	TOTALEXPENDITURE	TOTALINCOME
1	2012	3000	5950
1	2013	12000	12450
1	2014	8000	12250
2	2012	3400	4680
2	2013	3000	4350
2	2014	1000	2200
3	2012	500	640
3	2013	1420	1480
3	2014	2150	3620

Figure 21 – Monash Club Fact Table