[FIT5195 Major Assignment]

[Tutorial05_11]

HONGLIANG TANG (27135519)
ZIHAN WANG (28975987)

MONASH University
Information Technology

GROUP ASSIGNMENT COVER SHEET

Student ID Number	Surname	Given Names
28975987	Wang	Zihan
27135519	Tang	Hongliang

* Please include the names of all other group members.

Unit name and code	FIT5195 Business intelligence and data warehousing - S1 2021	
Title of assignment	Major Assignment	
Lecturer/tutor	David Taniar/Shuyi Sun, Joe Shao, Xiaojiao Du, David Cheng Zarate	
Tutorial day and time	Monday 10am-12pm	**Campus** Clayton

Is this an authorised group assignment? [✓ √ **Yes**] ☐ **No**

Has any part of this assignment been previously submitted as part of another unit/course? ☐ **Yes** [✓ √ **No**]

Due Date 05/05/2021	**Date submitted** 05/05/2021

All work must be submitted by the due date. If an extension of work is granted this must be specified with the signature of the lecturer/tutor.
Extension granted until (date) .... **Signature of lecturer/tutor** ....

Please note that it is your responsibility to retain copies of your assessments.

Intentional plagiarism or collusion amounts to cheating under Part 7 of the Monash University (Council) Regulations

Plagiarism: Plagiarism means taking and using another person's ideas or manner of expressing them and passing them off as one's own. For example, by failing to give appropriate acknowledgement. The material used can be from any source (staff, students or the internet, published and unpublished works).

Collusion: Collusion means unauthorised collaboration with another person on assessable written, oral or practical work and includes paying another person to complete all or part of the work.

Where there are reasonable grounds for believing that intentional plagiarism or collusion has occurred, this will be reported to the Associate Dean (Education) or delegate, who may disallow the work concerned by prohibiting assessment or refer the matter to the Faculty Discipline Panel for a hearing.

Student Statement:

- I have read the university's Student Academic Integrity Policy and Procedures.
- I understand the consequences of engaging in plagiarism and collusion as described in Part 7 of the Monash University (Council) Regulations http://adm.monash.edu/legal/legislation/statutes
- I have taken proper care to safeguard this work and made all reasonable efforts to ensure it could not be copied.
- No part of this assignment has been previously submitted as part of another unit/course.
- I acknowledge and agree that the assessor of this assignment may for the purposes of assessment, reproduce the assignment and:
  i. provide to another member of faculty and any external marker; and/or
  ii. submit it to a text matching software; and/or
  iii. submit it to a text matching software which may then retain a copy of the assignment on its database for the purpose of future plagiarism checking.
- I certify that I have not plagiarised the work of others or participated in unauthorised collaboration when preparing this assignment.

Signature .... *Date*..

* delete (iii) if not applicable

Signature [illegible] Date: 04/05/2021 Signature ________________ Date: ________

Signature [illegible] Date: 04/05/2021 Signature ________________ Date: ________

Signature ________________ Date: ________________ Signature ________________ Date: ________

Updated: 17 Jun 2014

Details of Oracle Account

Zihan Wang (28975987):

Username: S28975987
Password: student

HongLiang Tang (27135519):

Username: S27135519

Password: student

Contribution Declaration Form
(to be completed by all team members)

Please fill in the form with the contribution from each student towards the assignment.

1 NAME AND CONTRIBUTION DETAILS

Student ID	Student Name	Contribution Percentage
28975987	Zihan Wang	50%
27135519	HongLiang Tang	50%

2 DECLARATION

List of parts that each student did:

1. Zihan：
   - E/R diagram
   - Data Cleaning
   - Version-1 SQL
   - Report format

2. HongLiang:
   - E/R diagram
   - Data Cleaning
   - Version-2
   - SCD Type

We declare that:

- The information we have supplied in or with this form is complete and correct.
- We understand that the information we have provided in this form will be used for individual assessment of the assignment.

3 SIGNATURE

Signatures	[illegible]	[illegible]

	Day Month Year
Date	04 / 05/ 2021

FIT5195 Major Assignment

Tutorial05_11

HongLiang Tang (27135519)

Zihan Wang (28975987)

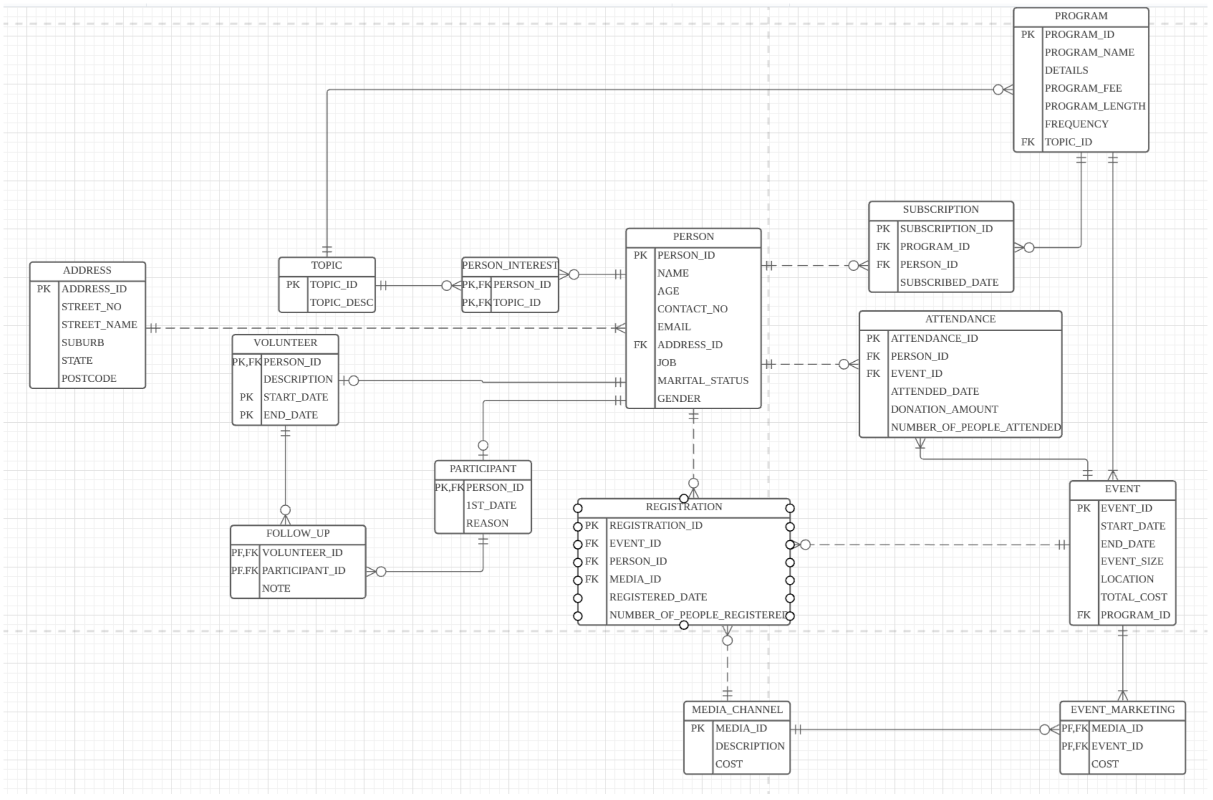
Table of Contents

C.1 Tasks

a) E/R Diagram

There are some basic assumptions made to make the E/R Diagram clearer.

1. A new discovered topic might not have a program yet.
2. a volunteer doesn't write survey, survey is given to normal participants and used to obtain participants interests on topics.
3. People may come to the expo but s/he didn't find any program to subscribe as they are not interested.
4. some "cold" events may have no person to register.
5. If there is a program, there will always be events organized for it.
6. an existing Event will need at least one media to promote so that people will know its existence. This will significantly reduce the possibility that nobody participates in an organized event.
7. When registered, participants will fill the media channel from which they got the related event information. It is reasonable to assume that not all media are involved in promotion. So, it means that a media can cause 0 or many registration
8. The media can promote 0 or many events. It means MonExplore won't choose all possible media on the market since they would like to control the cost as necessary as a NPO(non-profit organization).



b) Data cleaning process

We have performed data cleaning from the following aspects to remove dirty data:

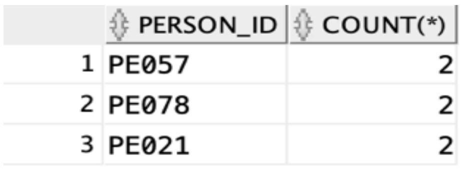
1) Duplication problems

Justification:

From this database, the duplicate records all have the identical information, it is not meaningful to keep multiple identical records therefore we remove the duplicate records and only keep one distinct record.

a) PERSON: Person_id

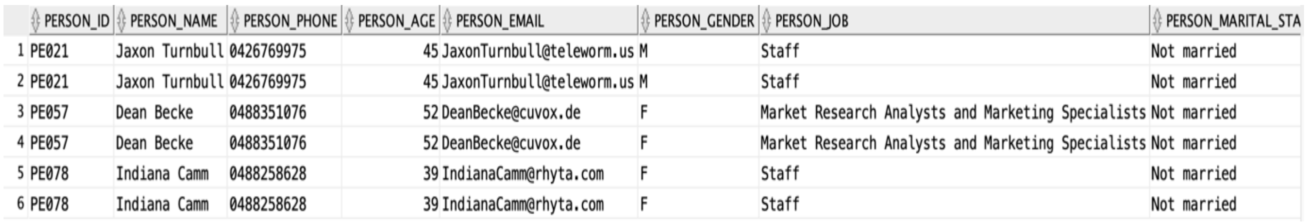
```
--PE057, PE078, PE021 have identical duplicate records

SELECT
   person_id,
   COUNT(*)
FROM
   person
GROUP BY
   person_id
HAVING
   COUNT(*) > 1;
```



	PERSON_ID	COUNT(*)
1	PE057	2
2	PE078	2
3	PE021	2

Before data cleaning:

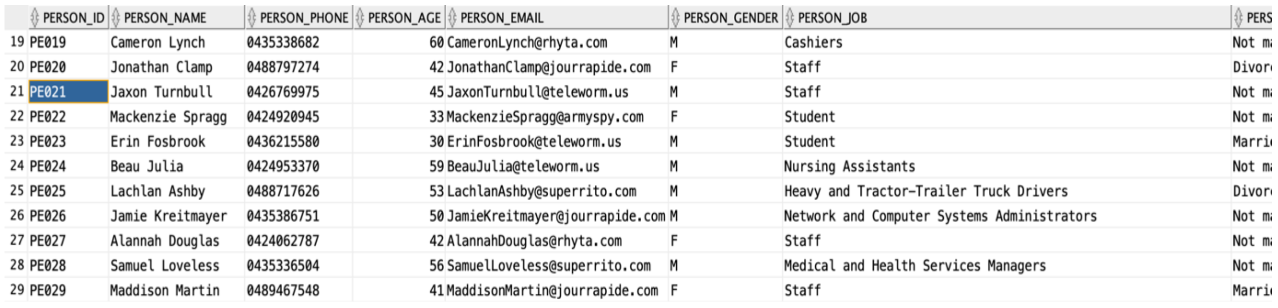
```
SELECT
   *
FROM
   person;
```



	PERSON_ID	PERSON_NAME	PERSON_PHONE	PERSON_AGE	PERSON_EMAIL	PERSON_GENDER	PERSON_JOB	PERSON_MARITAL_STA
1	PE021	Jaxon Turnbull	0426769975	45	JaxonTurnbull@teleworm.us	M	Staff	Not married
2	PE021	Jaxon Turnbull	0426769975	45	JaxonTurnbull@teleworm.us	M	Staff	Not married
3	PE057	Dean Becke	0488351076	52	DeanBecke@cuvox.de	F	Market Research Analysts and Marketing Specialists	Not married
4	PE057	Dean Becke	0488351076	52	DeanBecke@cuvox.de	F	Market Research Analysts and Marketing Specialists	Not married
5	PE078	Indiana Camm	0488258628	39	IndianaCamm@rhyta.com	F	Staff	Not married
6	PE078	Indiana Camm	0488258628	39	IndianaCamm@rhyta.com	F	Staff	Not married

After data cleaning:

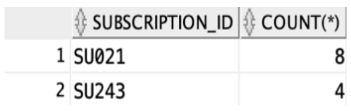
```
DROP TABLE person;

CREATE TABLE person
   AS
      SELECT DISTINCT
         *
      FROM
         monexplore.person;

SELECT
   *
FROM
   person;
```



	PERSON_ID	PERSON_NAME	PERSON_PHONE	PERSON_AGE	PERSON_EMAIL	PERSON_GENDER	PERSON_JOB	PERS
19	PE019	Cameron Lynch	0435338682	60	CameronLynch@rhyta.com	M	Cashiers	Not ma
20	PE020	Jonathan Clamp	0488797274	42	JonathanClamp@jourrapide.com	F	Staff	Divor
21	PE021	Jaxon Turnbull	0426769975	45	JaxonTurnbull@teleworm.us	M	Staff	Not ma
22	PE022	Mackenzie Spragg	0424920945	33	MackenzieSpragg@armyspy.com	F	Student	Not ma
23	PE023	Erin Fosbrook	0436215580	30	ErinFosbrook@teleworm.us	M	Student	Marri
24	PE024	Beau Julia	0424953370	59	BeauJulia@teleworm.us	M	Nursing Assistants	Not ma
25	PE025	Lachlan Ashby	0488717626	53	LachlanAshby@superrito.com	M	Heavy and Tractor-Trailer Truck Drivers	Divor
26	PE026	Jamie Kreitmayer	0435386751	50	JamieKreitmayer@jourrapide.com	M	Network and Computer Systems Administrators	Not ma
27	PE027	Alannah Douglas	0424062787	42	AlannahDouglas@rhyta.com	F	Staff	Not ma
28	PE028	Samuel Loveless	0435336504	56	SamuelLoveless@superrito.com	M	Medical and Health Services Managers	Not ma
29	PE029	Maddison Martin	0489467548	41	MaddisonMartin@jourrapide.com	F	Staff	Marri

b) SUBSCRIPTION: Subscription_id

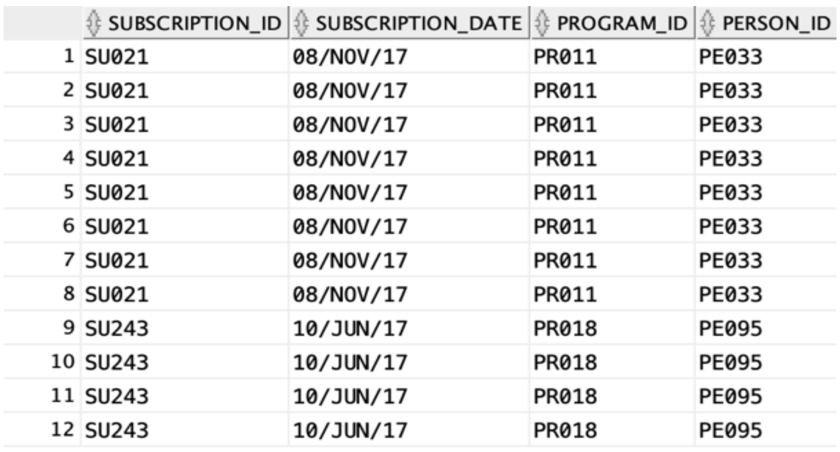
```
--SU021, SU243 have identical duplicate records
SELECT
         subscription_id,
         COUNT(*)
FROM
         subscription
GROUP BY
         subscription_id
HAVING
          COUNT(*) > 1;
```



	SUBSCRIPTION_ID	COUNT(*)
1	SU021	8
2	SU243	4

Before data cleaning:

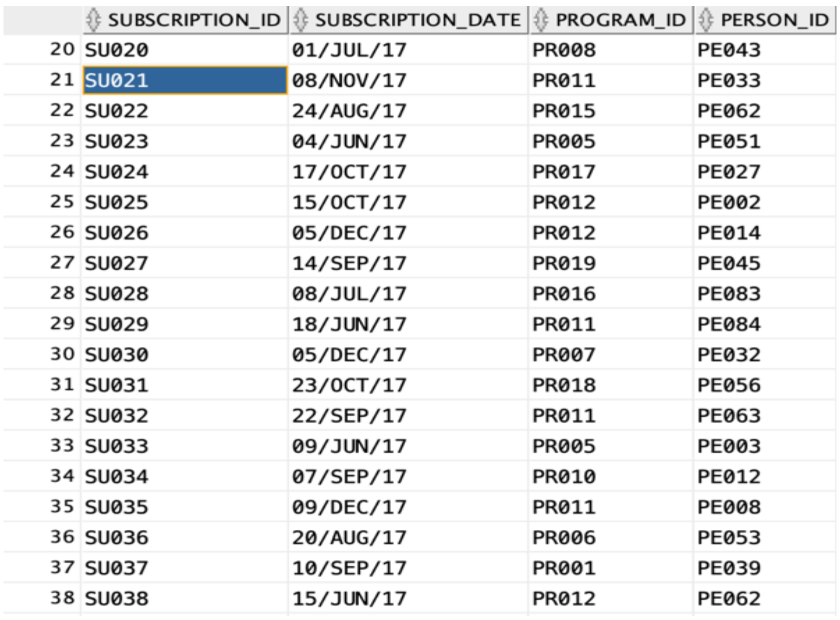
```
SELECT
   *
FROM
   subscription
WHERE
   subscription_id = 'SU021'
   OR subscription_id = 'SU243';
```



	SUBSCRIPTION_ID	SUBSCRIPTION_DATE	PROGRAM_ID	PERSON_ID
1	SU021	08/NOV/17	PR011	PE033
2	SU021	08/NOV/17	PR011	PE033
3	SU021	08/NOV/17	PR011	PE033
4	SU021	08/NOV/17	PR011	PE033
5	SU021	08/NOV/17	PR011	PE033
6	SU021	08/NOV/17	PR011	PE033
7	SU021	08/NOV/17	PR011	PE033
8	SU021	08/NOV/17	PR011	PE033
9	SU243	10/JUN/17	PR018	PE095
10	SU243	10/JUN/17	PR018	PE095
11	SU243	10/JUN/17	PR018	PE095
12	SU243	10/JUN/17	PR018	PE095

After data cleaning:

```
DROP TABLE subscription;

CREATE TABLE subscription
   AS
      SELECT DISTINCT
         *
      FROM
         monexplore.subscription;

SELECT
   *
FROM
   subscription
ORDER BY
   subscription_id;
```



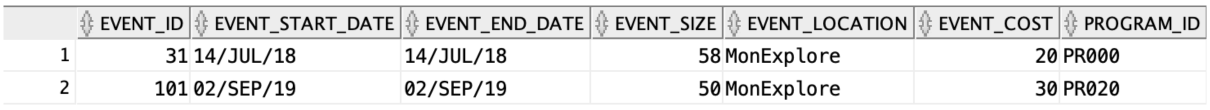
	SUBSCRIPTION_ID	SUBSCRIPTION_DATE	PROGRAM_ID	PERSON_ID
20	SU020	01/JUL/17	PR008	PE043
21	SU021	08/NOV/17	PR011	PE033
22	SU022	24/AUG/17	PR015	PE062
23	SU023	04/JUN/17	PR005	PE051
24	SU024	17/OCT/17	PR017	PE027
25	SU025	15/OCT/17	PR012	PE002
26	SU026	05/DEC/17	PR012	PE014
27	SU027	14/SEP/17	PR019	PE045
28	SU028	08/JUL/17	PR016	PE083
29	SU029	18/JUN/17	PR011	PE084
30	SU030	05/DEC/17	PR007	PE032
31	SU031	23/OCT/17	PR018	PE056
32	SU032	22/SEP/17	PR011	PE063
33	SU033	09/JUN/17	PR005	PE003
34	SU034	07/SEP/17	PR010	PE012
35	SU035	09/DEC/17	PR011	PE008
36	SU036	20/AUG/17	PR006	PE053
37	SU037	10/SEP/17	PR001	PE039
38	SU038	15/JUN/17	PR012	PE062

2) Relationship problems

a) PROGRAM → EVENT

Before data cleaning:

```
--PR000 for event_id 31 and PR020 for event_id 101 do not exist
SELECT
    *
FROM
    event
WHERE
    program_id NOT IN (
        SELECT
            program_id
        FROM
         program
 );
```



	EVENT_ID	EVENT_START_DATE	EVENT_END_DATE	EVENT_SIZE	EVENT_LOCATION	EVENT_COST	PROGRAM_ID
1	31	14/JUL/18	14/JUL/18	58	MonExplore	20	PR000
2	101	02/SEP/19	02/SEP/19	50	MonExplore	30	PR020

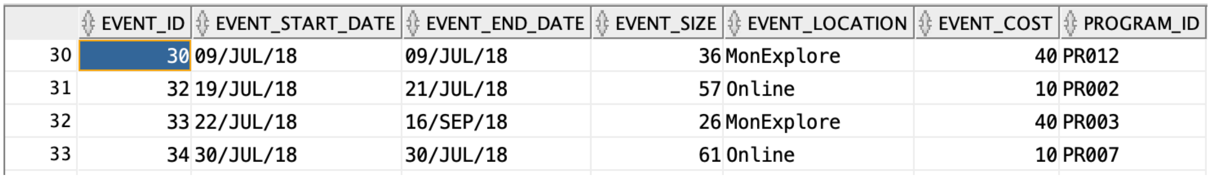
Justification:

Program_id PR000 for Event_id 31 and Program_id PR020 for Event_id 101 both do not exist in the Program table. It is not possible to attend an event of a non-existing program. Therefore, we delete all relevant data from EVENT, EVENT_MARKETING, ATTENDANCE and REGISTRATION where event_id = 31 or event_id = 101.

After data cleaning:

```
DELETE FROM event
WHERE
    event_id = 31
    OR event_id = 101;

DELETE FROM event_marketing
WHERE
    event_id = 31
    OR event_id = 101;

DELETE FROM attendance
WHERE
    event_id = 31
    OR event_id = 101;

DELETE FROM registration
WHERE
    event_id = 31
   OR event_id = 101;
```

```
COMMIT;

SELECT
*
 FROM
event
ORDER BY
   event_id;
```

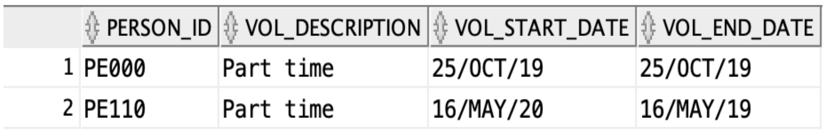


	EVENT_ID	EVENT_START_DATE	EVENT_END_DATE	EVENT_SIZE	EVENT_LOCATION	EVENT_COST	PROGRAM_ID
30	30	09/JUL/18	09/JUL/18	36	MonExplore	40	PR012
31	32	19/JUL/18	21/JUL/18	57	Online	10	PR002
32	33	22/JUL/18	16/SEP/18	26	MonExplore	40	PR003
33	34	30/JUL/18	30/JUL/18	61	Online	10	PR007

b) PERSON → VOLUNTEER

Before data cleaning:

```
--PE000 and PE110 do not exist
SELECT
   *
FROM
   volunteer
WHERE
   person_id NOT IN (
      SELECT
         person_id
      FROM
         person
   );
```



	PERSON_ID	VOL_DESCRIPTION	VOL_START_DATE	VOL_END_DATE
1	PE000	Part time	25/OCT/19	25/OCT/19
2	PE110	Part time	16/MAY/20	16/MAY/19

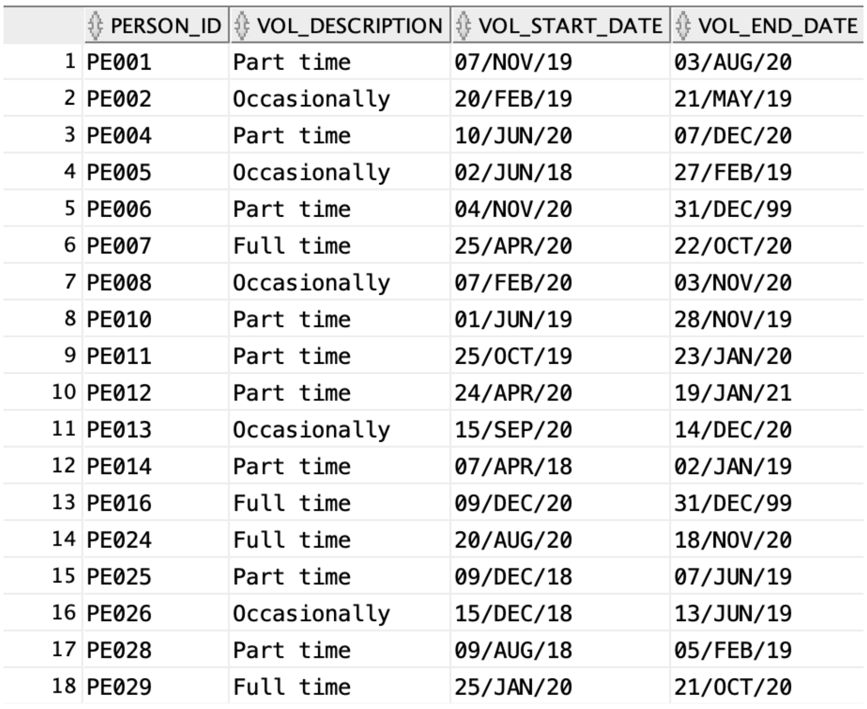
Justification:

As there is no record of Person_id = PE000 or PE110 in another related table(follow_up), we can safely delete these two records.

After data cleaning:

```
DELETE FROM volunteer
WHERE
   person_id = 'PE000'
   OR person_id = 'PE110';
```

```
DELETE FROM follow_up
WHERE
    person_id = 'PE000'
   OR person_id = 'PE110';
COMMIT;
```



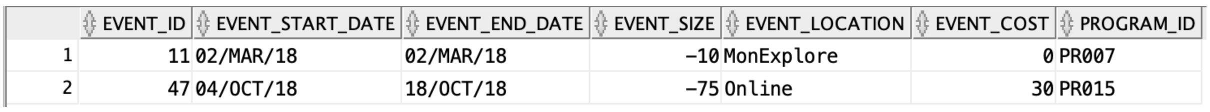
	PERSON_ID	VOL_DESCRIPTION	VOL_START_DATE	VOL_END_DATE
1	PE001	Part time	07/NOV/19	03/AUG/20
2	PE002	Occasionally	20/FEB/19	21/MAY/19
3	PE004	Part time	10/JUN/20	07/DEC/20
4	PE005	Occasionally	02/JUN/18	27/FEB/19
5	PE006	Part time	04/NOV/20	31/DEC/99
6	PE007	Full time	25/APR/20	22/OCT/20
7	PE008	Occasionally	07/FEB/20	03/NOV/20
8	PE010	Part time	01/JUN/19	28/NOV/19
9	PE011	Part time	25/OCT/19	23/JAN/20
10	PE012	Part time	24/APR/20	19/JAN/21
11	PE013	Occasionally	15/SEP/20	14/DEC/20
12	PE014	Part time	07/APR/18	02/JAN/19
13	PE016	Full time	09/DEC/20	31/DEC/99
14	PE024	Full time	20/AUG/20	18/NOV/20
15	PE025	Part time	09/DEC/18	07/JUN/19
16	PE026	Occasionally	15/DEC/18	13/JUN/19
17	PE028	Part time	09/AUG/18	05/FEB/19
18	PE029	Full time	25/JAN/20	21/OCT/20

3) Inconsistent and incorrect values

a) EVENT: Event_size less than zero

Before data cleaning:

```
-----check if size < 0
--Event_id 11 and 47 have event_size < 0
SELECT
    *
FROM
    event
WHERE
    event_size <= 0;
```



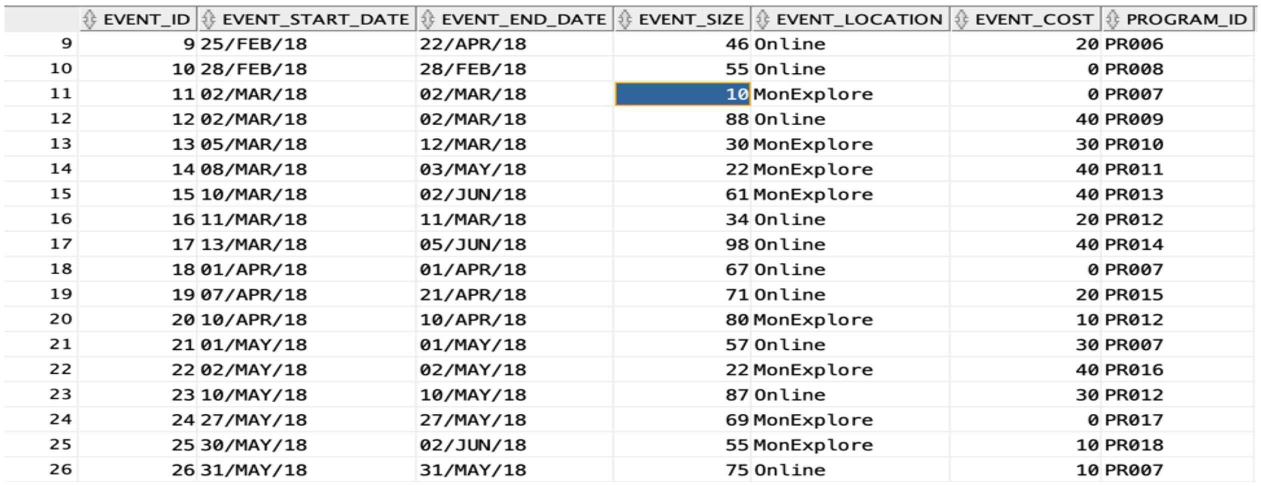
	EVENT_ID	EVENT_START_DATE	EVENT_END_DATE	EVENT_SIZE	EVENT_LOCATION	EVENT_COST	PROGRAM_ID
1	11	02/MAR/18	02/MAR/18	-10	MonExplore	0	PR007
2	47	04/OCT/18	18/OCT/18	-75	Online	30	PR015

Justification:

By comparing with data from Registration table, we found out that there was a typing mistake for the Event_size, we changed the value to positive number.

After data cleaning:

```
UPDATE event
SET
   event_size = 10
WHERE
   event_id = 11;

UPDATE event
SET
   event_size = 75
WHERE
   event_id = 47;

COMMIT;
--view event table
SELECT
   *
FROM
   event;
```

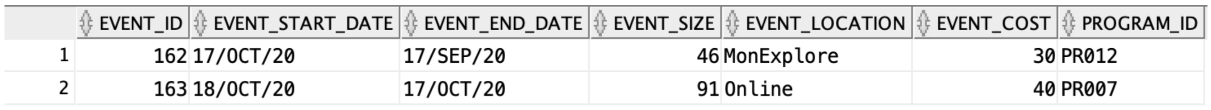


	EVENT_ID	EVENT_START_DATE	EVENT_END_DATE	EVENT_SIZE	EVENT_LOCATION	EVENT_COST	PROGRAM_ID
9	9	25/FEB/18	22/APR/18	46	Online	20	PR006
10	10	28/FEB/18	28/FEB/18	55	Online	0	PR008
11	11	02/MAR/18	02/MAR/18	10	MonExplore	0	PR007
12	12	02/MAR/18	02/MAR/18	88	Online	40	PR009
13	13	05/MAR/18	12/MAR/18	30	MonExplore	30	PR010
14	14	08/MAR/18	03/MAY/18	22	MonExplore	40	PR011
15	15	10/MAR/18	02/JUN/18	61	MonExplore	40	PR013
16	16	11/MAR/18	11/MAR/18	34	Online	20	PR012
17	17	13/MAR/18	05/JUN/18	98	Online	40	PR014
18	18	01/APR/18	01/APR/18	67	Online	0	PR007
19	19	07/APR/18	21/APR/18	71	Online	20	PR015
20	20	10/APR/18	10/APR/18	80	MonExplore	10	PR012
21	21	01/MAY/18	01/MAY/18	57	Online	30	PR007
22	22	02/MAY/18	02/MAY/18	22	MonExplore	40	PR016
23	23	10/MAY/18	10/MAY/18	87	Online	30	PR012
24	24	27/MAY/18	27/MAY/18	69	MonExplore	0	PR017
25	25	30/MAY/18	02/JUN/18	55	MonExplore	10	PR018
26	26	31/MAY/18	31/MAY/18	75	Online	10	PR007

b) EVENT: event_start_date > event_end date

Before data cleaning:

```
--check if event_start_date > event_end_date
--event_id 162 and 163
--focus on date first
SELECT
   event_id,
   to_char(event_start_date, 'YYYY-MM-DD HH24:MI:SS'),
   to_char(event_end_date, 'YYYY-MM-DD HH24:MI:SS')
FROM
   event
WHERE
       to_char(event_start_date, 'YYYY-MM-DD') > to_char(event_end_date,
       'YYYY-MM-DD');
```



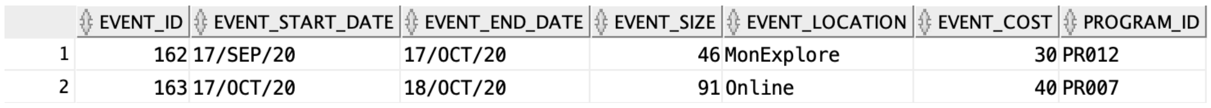
	EVENT_ID	EVENT_START_DATE	EVENT_END_DATE	EVENT_SIZE	EVENT_LOCATION	EVENT_COST	PROGRAM_ID
1	162	17/OCT/20	17/SEP/20	46	MonExplore	30	PR012
2	163	18/OCT/20	17/OCT/20	91	Online	40	PR007

Justification:

Event_id 162 and 163 has conflict dates. As these records are related to records in registration, we cannot delete them. We can assume the inserted value are mistakenly reversed, so we need to reverse them back.

After data cleaning:

```
--After data cleaning
UPDATE event
SET
   event_start_date = event_end_date,
   event_end_date = event_start_date
WHERE
   event_id = 162
       OR event_id = 163;
    COMMIT;
```

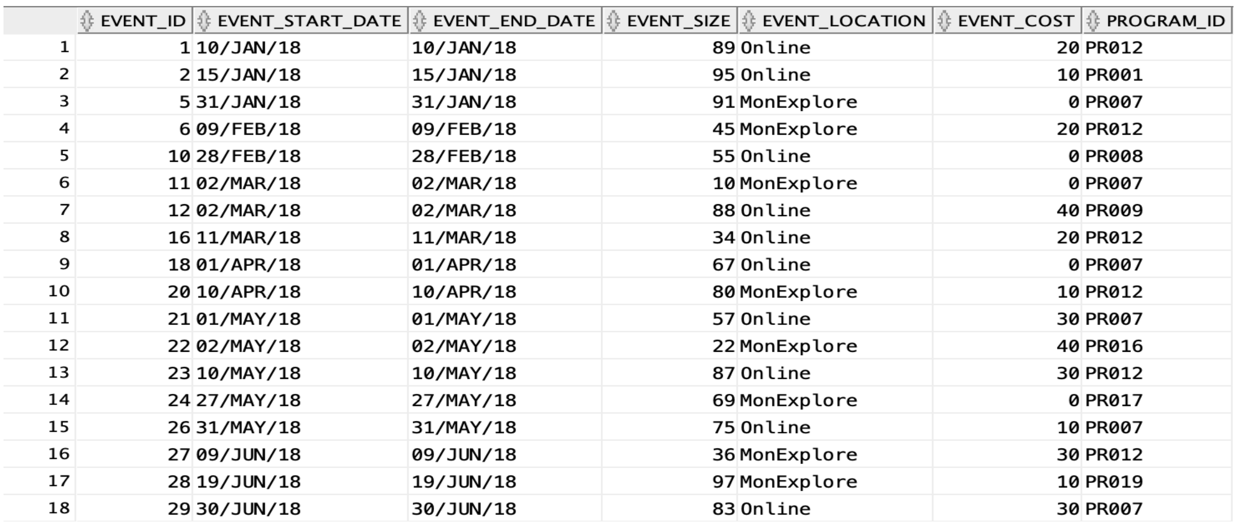


	EVENT_ID	EVENT_START_DATE	EVENT_END_DATE	EVENT_SIZE	EVENT_LOCATION	EVENT_COST	PROGRAM_ID
1	162	17/SEP/20	17/OCT/20	46	MonExplore	30	PR012
2	163	17/OCT/20	18/OCT/20	91	Online	40	PR007

c) EVENT: For the same day time conflict

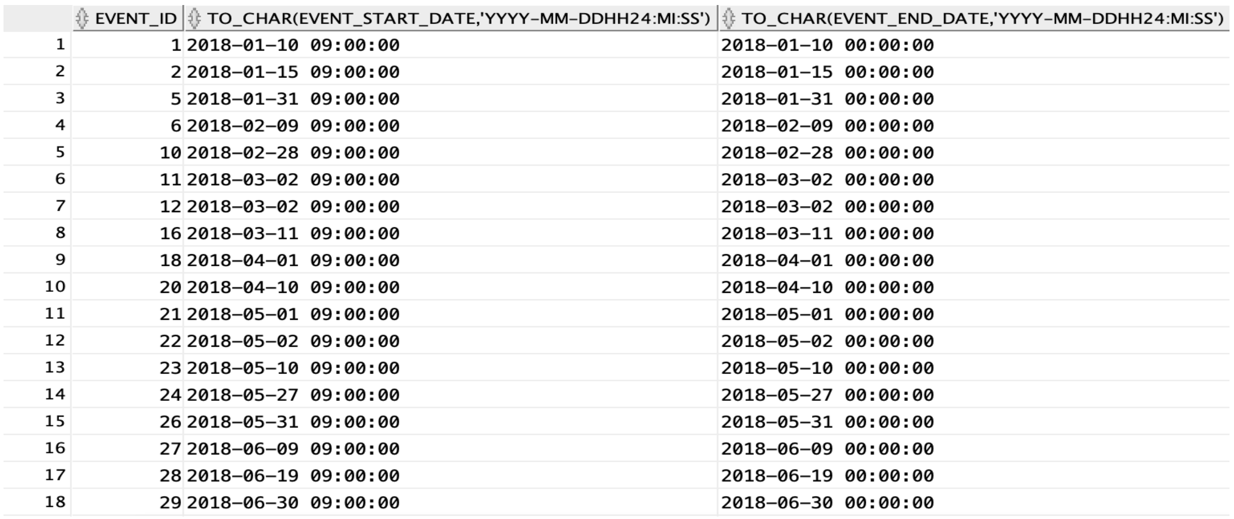
Before data cleaning:

```
/*now focus on time with the same date*/
SELECT
   *
FROM
   event
WHERE
   to_char(event_start_date, 'YYYY-MM-DD HH24:MI:SS') >
to_char(event_end_date, 'YYYY-MM-DD HH24:MI:SS');
```



	EVENT_ID	EVENT_START_DATE	EVENT_END_DATE	EVENT_SIZE	EVENT_LOCATION	EVENT_COST	PROGRAM_ID
1	1	10/JAN/18	10/JAN/18	89	Online	20	PR012
2	2	15/JAN/18	15/JAN/18	95	Online	10	PR001
3	5	31/JAN/18	31/JAN/18	91	MonExplore	0	PR007
4	6	09/FEB/18	09/FEB/18	45	MonExplore	20	PR012
5	10	28/FEB/18	28/FEB/18	55	Online	0	PR008
6	11	02/MAR/18	02/MAR/18	10	MonExplore	0	PR007
7	12	02/MAR/18	02/MAR/18	88	Online	40	PR009
8	16	11/MAR/18	11/MAR/18	34	Online	20	PR012
9	18	01/APR/18	01/APR/18	67	Online	0	PR007
10	20	10/APR/18	10/APR/18	80	MonExplore	10	PR012
11	21	01/MAY/18	01/MAY/18	57	Online	30	PR007
12	22	02/MAY/18	02/MAY/18	22	MonExplore	40	PR016
13	23	10/MAY/18	10/MAY/18	87	Online	30	PR012
14	24	27/MAY/18	27/MAY/18	69	MonExplore	0	PR017
15	26	31/MAY/18	31/MAY/18	75	Online	10	PR007
16	27	09/JUN/18	09/JUN/18	36	MonExplore	30	PR012
17	28	19/JUN/18	19/JUN/18	97	MonExplore	10	PR019
18	29	30/JUN/18	30/JUN/18	83	Online	30	PR007

```
--view the exact event time
SELECT
    event_id,
    to_char(event_start_date, 'YYYY-MM-DD HH24:MI:SS'),
    to_char(event_end_date, 'YYYY-MM-DD HH24:MI:SS')
FROM
    (
        SELECT
            *
        FROM
            event
        WHERE
            event_start_date > event_end_date
    );
```



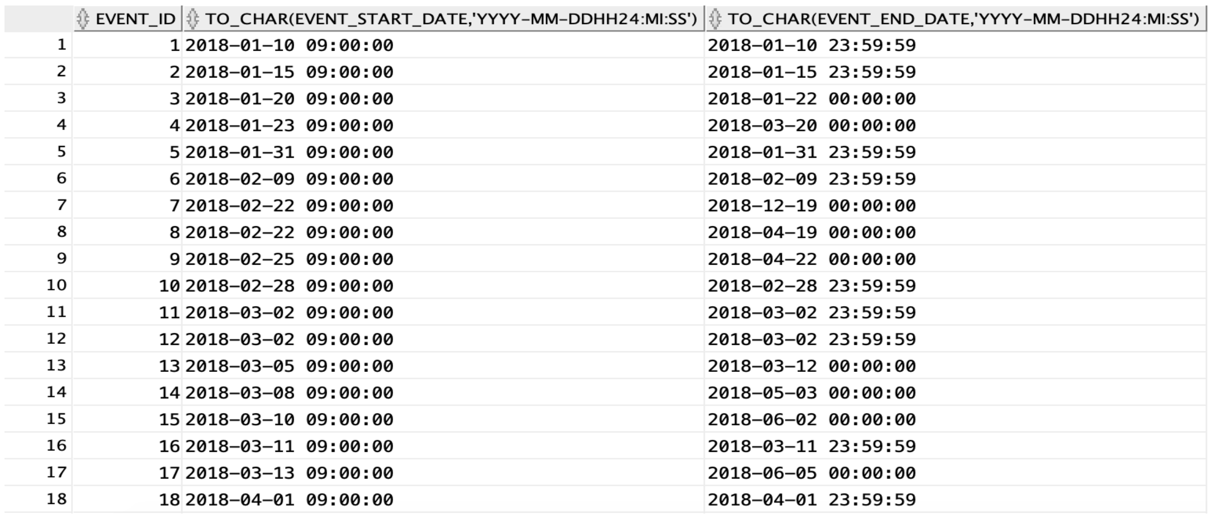
	EVENT_ID	TO_CHAR(EVENT_START_DATE,'YYYY-MM-DDHH24:MI:SS')	TO_CHAR(EVENT_END_DATE,'YYYY-MM-DDHH24:MI:SS')
1	1	2018-01-10 09:00:00	2018-01-10 00:00:00
2	2	2018-01-15 09:00:00	2018-01-15 00:00:00
3	5	2018-01-31 09:00:00	2018-01-31 00:00:00
4	6	2018-02-09 09:00:00	2018-02-09 00:00:00
5	10	2018-02-28 09:00:00	2018-02-28 00:00:00
6	11	2018-03-02 09:00:00	2018-03-02 00:00:00
7	12	2018-03-02 09:00:00	2018-03-02 00:00:00
8	16	2018-03-11 09:00:00	2018-03-11 00:00:00
9	18	2018-04-01 09:00:00	2018-04-01 00:00:00
10	20	2018-04-10 09:00:00	2018-04-10 00:00:00
11	21	2018-05-01 09:00:00	2018-05-01 00:00:00
12	22	2018-05-02 09:00:00	2018-05-02 00:00:00
13	23	2018-05-10 09:00:00	2018-05-10 00:00:00
14	24	2018-05-27 09:00:00	2018-05-27 00:00:00
15	26	2018-05-31 09:00:00	2018-05-31 00:00:00
16	27	2018-06-09 09:00:00	2018-06-09 00:00:00
17	28	2018-06-19 09:00:00	2018-06-19 00:00:00
18	29	2018-06-30 09:00:00	2018-06-30 00:00:00

Justification:

It seems the staff mistakenly thought 00:00:00 as 24:00:00. we cannot input 24:00:00 as the last moment of a day, but we can correct it as 23:59:59.

After data cleaning:

```
UPDATE event
SET
   event_end_date = to_date((to_char(event_end_date, 'YYYY-MM-DD')
                      || ' 23:59:59'), 'YYYY-MM-DD HH24:MI:SS')
WHERE
   event_start_date > event_end_date;

SELECT
   event_id,
   to_char(event_start_date, 'YYYY-MM-DD HH24:MI:SS'),
   to_char(event_end_date, 'YYYY-MM-DD HH24:MI:SS')
FROM
   event;
/* now all the date has been corrected*/
COMMIT;
```

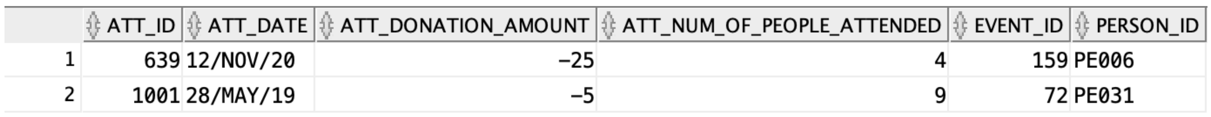


	EVENT_ID	TO_CHAR(EVENT_START_DATE,'YYYY-MM-DDHH24:MI:SS')	TO_CHAR(EVENT_END_DATE,'YYYY-MM-DDHH24:MI:SS')
1	1	2018-01-10 09:00:00	2018-01-10 23:59:59
2	2	2018-01-15 09:00:00	2018-01-15 23:59:59
3	3	2018-01-20 09:00:00	2018-01-22 00:00:00
4	4	2018-01-23 09:00:00	2018-03-20 00:00:00
5	5	2018-01-31 09:00:00	2018-01-31 23:59:59
6	6	2018-02-09 09:00:00	2018-02-09 23:59:59
7	7	2018-02-22 09:00:00	2018-12-19 00:00:00
8	8	2018-02-22 09:00:00	2018-04-19 00:00:00
9	9	2018-02-25 09:00:00	2018-04-22 00:00:00
10	10	2018-02-28 09:00:00	2018-02-28 23:59:59
11	11	2018-03-02 09:00:00	2018-03-02 23:59:59
12	12	2018-03-02 09:00:00	2018-03-02 23:59:59
13	13	2018-03-05 09:00:00	2018-03-12 00:00:00
14	14	2018-03-08 09:00:00	2018-05-03 00:00:00
15	15	2018-03-10 09:00:00	2018-06-02 00:00:00
16	16	2018-03-11 09:00:00	2018-03-11 23:59:59
17	17	2018-03-13 09:00:00	2018-06-05 00:00:00
18	18	2018-04-01 09:00:00	2018-04-01 23:59:59

d) ATTENDANCE: att_donation_amount < 0

Before data cleaning:

```
--check for money < 0
--att_id 639 and 1001
SELECT
   *
FROM
   attendance
WHERE
   att_donation_amount < 0;
```



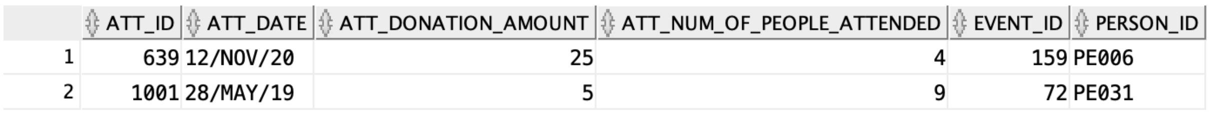
	ATT_ID	ATT_DATE	ATT_DONATION_AMOUNT	ATT_NUM_OF_PEOPLE_ATTENDED	EVENT_ID	PERSON_ID
1	639	12/NOV/20	-25	4	159	PE006
2	1001	28/MAY/19	-5	9	72	PE031

Justification:

Money cannot be negative value; we correct it to a positive value.

After data cleaning:

```
UPDATE attendance
SET
   att_donation_amount = 25
WHERE
   att_id = 639;

UPDATE attendance
SET
   att_donation_amount = 5
WHERE
       att_id = 1001;
COMMIT;
--view updated value
SELECT
   *
FROM
   attendance
WHERE
   att_id = 639
   OR att_id = 1001;
```



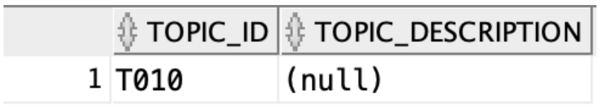
	ATT_ID	ATT_DATE	ATT_DONATION_AMOUNT	ATT_NUM_OF_PEOPLE_ATTENDED	EVENT_ID	PERSON_ID
1	639	12/NOV/20	25	4	159	PE006
2	1001	28/MAY/19	5	9	72	PE031

4) The null value problems

a) TOPIC: topic_id is null

Before data cleaning:

```
SELECT
   *
FROM
   topic
WHERE
   topic_description IS NULL;
```



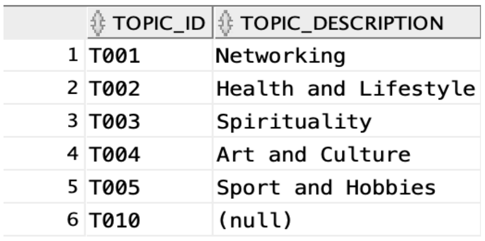
	TOPIC_ID	TOPIC_DESCRIPTION
1	T010	(null)

Justification:

There is no program related to topic T010, but there are two people interested in topic_id T010. We don't have to delete this topic since topic_description is not a pk and it is critical for the person_interest table. The topic may be newly created, and description is not updated in the system

After data cleaning:

```
--view topic
SELECT
   *
FROM
       topic;
```

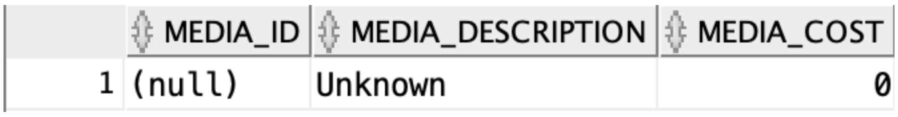


	TOPIC_ID	TOPIC_DESCRIPTION
1	T001	Networking
2	T002	Health and Lifestyle
3	T003	Spirituality
4	T004	Art and Culture
5	T005	Sport and Hobbies
6	T010	(null)

b) MEDIA_CHANNEL: media_id, media_description, media_cost are null

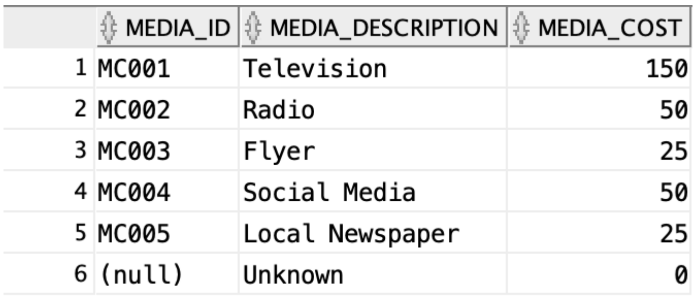
Before data cleaning:

```
SELECT
   *
FROM
   media_channel
WHERE
  media_id IS NULL;
```



	MEDIA_ID	MEDIA_DESCRIPTION	MEDIA_COST
1	(null)	Unknown	0

```
--view media_channel
SELECT
   *
FROM
   media_channel;
DELETE FROM media_channel
WHERE
       media_id IS NULL;
```



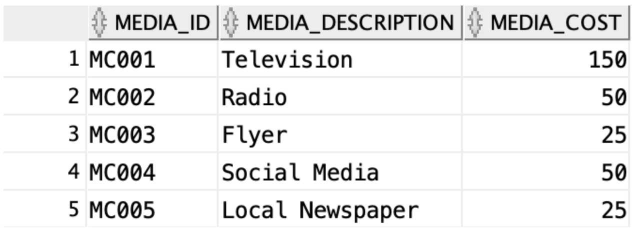
	MEDIA_ID	MEDIA_DESCRIPTION	MEDIA_COST
1	MC001	Television	150
2	MC002	Radio	50
3	MC003	Flyer	25
4	MC004	Social Media	50
5	MC005	Local Newspaper	25
6	(null)	Unknown	0

Justification:

This row of null value has no meaning, we can just delete it.

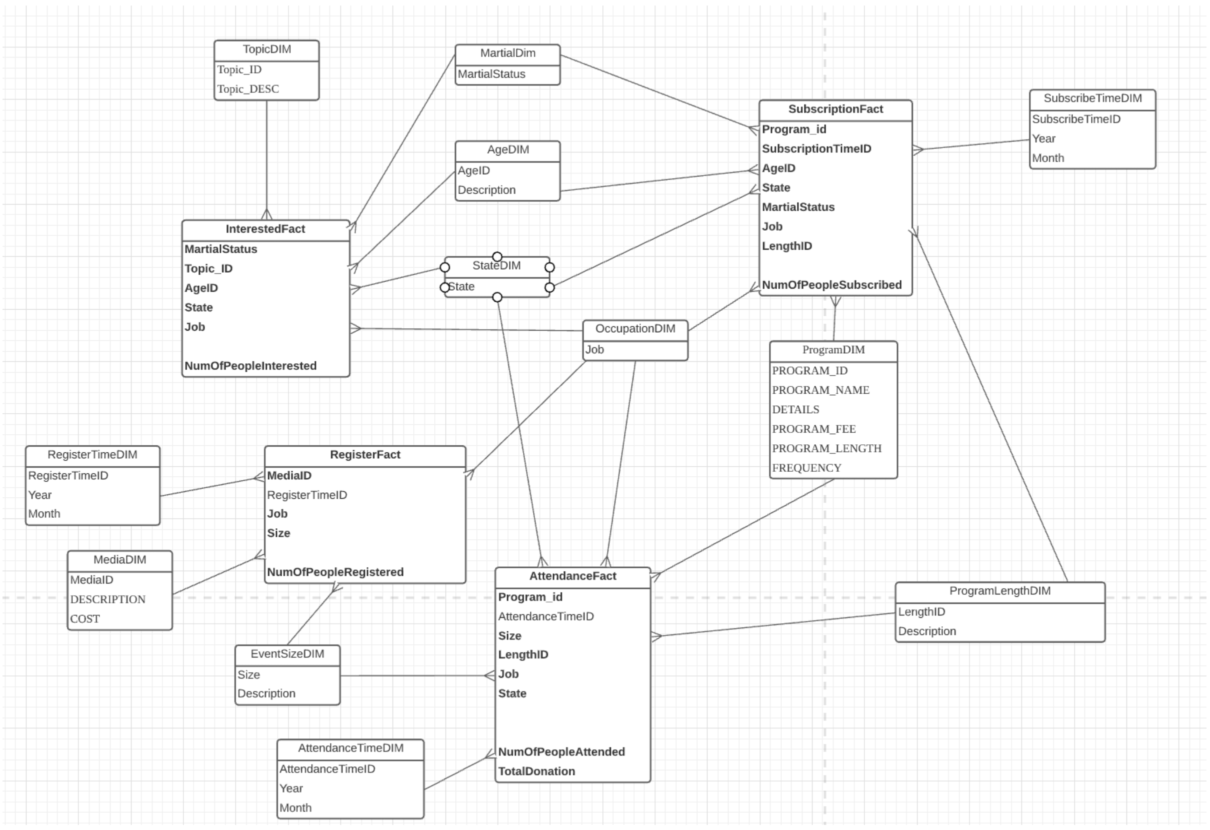
After data cleaning:

```
DELETE FROM media_channel
WHERE
        media_id IS NULL;
COMMIT;
--view media_channel
SELECT
    *
FROM
    media_channel;
```

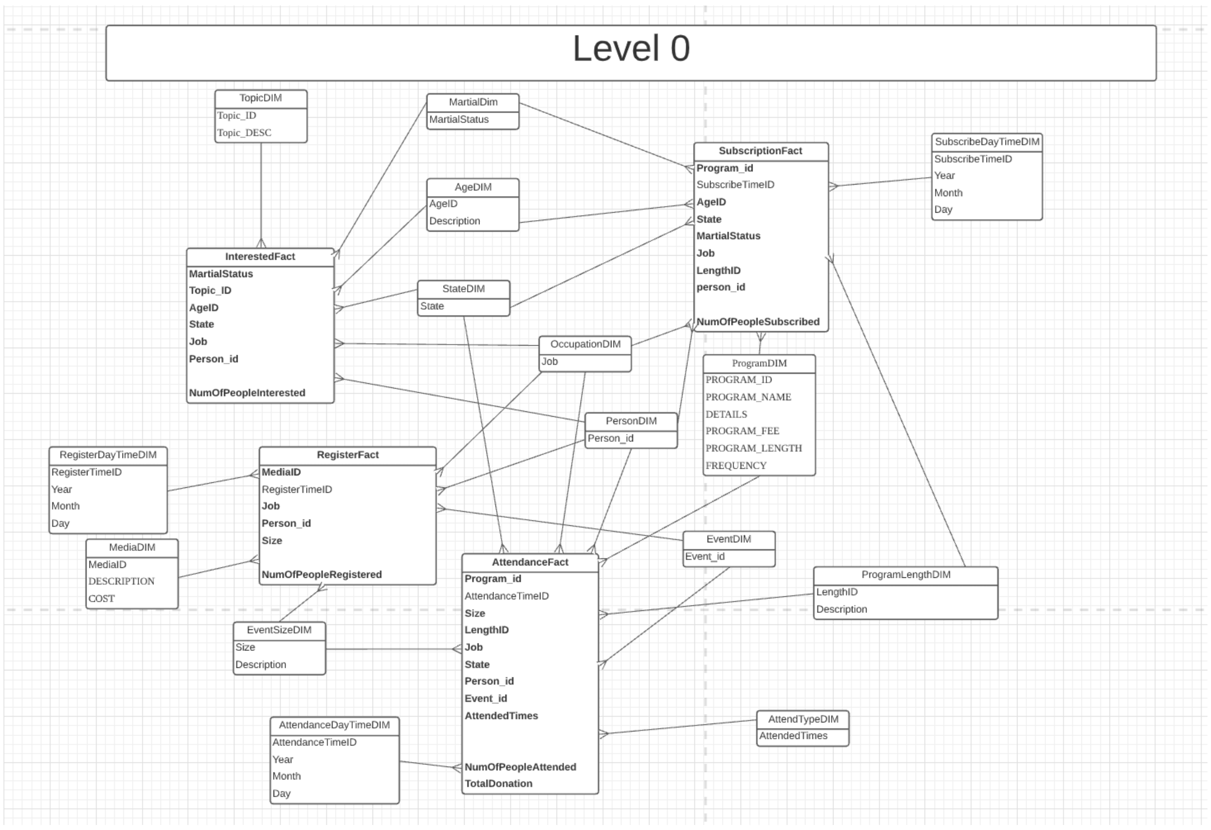


	MEDIA_ID	MEDIA_DESCRIPTION	MEDIA_COST
1	MC001	Television	150
2	MC002	Radio	50
3	MC003	Flyer	25
4	MC004	Social Media	50
5	MC005	Local Newspaper	25

c) Star/snowflake schema diagram V1



Star/snowflake schema diagram V2



d) SCD type

REGISTRATION:
Here, we know that a person can register multiple times due to some reasons, for example, he forgot he registered, or he want to change the original registration by making a new registration.
We also know that there are many mismatches in media_id between table registration and event_marketing. So, it would be reasonable for us to assume that people are possibly making mistake in filling their registration form by providing the wrong media source, and we cannot figure out from which real media channel this person acknowledged this event. What we do know is that this person did want to change his registration for that particular event.
Nevertheless, here we only care about how this person want to change his registration for a particular event, so we will always take the newest registration as the presented one, so we use SCD1 which reflection the most updated value as time changes.

e) Differences among two versions of star/snowflake schema

Explanation on the difference between Star schema version 1 and 2
In version 1, we have fact tables with highly aggregated value, we design the DW in such a way that it can fulfill the business requirements and questions. For example, interestfact to answer questions related to interest, attendancefact to answer questions related to attendance and donation etc.

The first difference between version 1 and version 2 is that the dimensions in version 2 have a higher granularity than the corresponding dimension in version 1.

Dimensions are the way we look at our fact table. In version 1, we have dimensions with relatively lower granularity than that of dimensions in version 2. For instance, the time dimensions in version 1 only identifies the year and month while the time dimensions in version can identify year, month and day. More detail means higher level of granularity, correspondingly, the related fact table will have high granularity as we look at it in a more detailed dimension. This is also one of the two ways to lower the level of aggregation of a star schema, it replaces an existing dimension with a higher granularity dimension.

The second difference between two versions is that we added new dimensions in version 2. For example, we added person and event dimension to version 2 to make each value in the related fact measure more granular. It means the value in fact measure will be broken down into more details on each record of the new dimension.

Understanding on the difference between different levels of aggregation

We know that higher level of aggregation means the fact measure is a precomputed aggregate value, this makes retrieval easy and efficient and it can assist decision-making processes. It becomes a problem when we want to drill know to find more information while the fact table does not store more detailed information. This is why the data warehousing granularity was introduced, and why we have levels of aggregation.

Level-0 has the highest level of granularity where no aggregation exists. it means that all domain tables are incorporated as dimension tables and there is no grouping in the dimension table. When we look at a star schema, we can only decide whether it is Level-0 or not, we cannot decide whether it is level 1 or level 2 or level 3, it depends on how many none level-0 below. The general rule is that the lower the level of granularity, the higher the level of aggregation. Therefore, we can only say Level (x+1) is more aggregated than level x star schema and the only difference between levels is their data granularity or aggregation level.

C.2 Tasks

a) SQL statements Version-1

```
---------------------------------------------------------------------------
--construct star/snowflake schema
---------------------------------------------------------------------------
--Version 1 High aggregation (Level 2)

--topicDIM
DROP TABLE topicdim;

CREATE TABLE topicdim
   AS
      SELECT
         *
      FROM
         topic;

SELECT
   *
FROM
   topicdim;

--MediaDIM
DROP TABLE mediadim;

CREATE TABLE mediadim
   AS
      SELECT
         *
      FROM
         media_channel;

SELECT
   *
FROM
   mediadim;

--MaritalDIM
DROP TABLE maritaldim;

CREATE TABLE maritaldim
   AS
      SELECT DISTINCT
         person_marital_status
      FROM
```

```
         person;

SELECT
    *
FROM
    maritaldim;

--StateDIM
DROP TABLE statedim;

CREATE TABLE statedim
    AS
        SELECT DISTINCT
            address_state
        FROM
            address;

SELECT
    *
FROM
    statedim;

--AgeGroupDIM
DROP TABLE agegroupdim;

CREATE TABLE agedim (
    age_id       VARCHAR(30),
    description  VARCHAR(50)
);

INSERT INTO agedim VALUES (
    'Child',
    '0-16'
);

INSERT INTO agedim VALUES (
    'Young Adults',
    '17-30'
);

INSERT INTO agedim VALUES (
    'Middle-aged Adults',
    '31-45'
);

INSERT INTO agedim VALUES (
    'Old-aged adults',
    '> 45'
);
```

```
SELECT
    *
FROM
    agedim;

--OccupationDIM
DROP TABLE occupationdim;

CREATE TABLE occupationdim (
    job VARCHAR(20)
);

INSERT INTO occupationdim VALUES ( 'Student' );

INSERT INTO occupationdim VALUES ( 'Staff' );

INSERT INTO occupationdim VALUES ( 'Community' );

SELECT
    *
FROM
    occupationdim;

--ProgramLengthDIM
DROP TABLE programlengthdim;

CREATE TABLE programlengthdim (
    length_id    VARCHAR(20),
    description  VARCHAR(20)
);

INSERT INTO programlengthdim VALUES (
    'Short Event',
    '< 3 sessions'
);

INSERT INTO programlengthdim VALUES (
    'Medium Event',
    '3 - 6 sessions'
);

INSERT INTO programlengthdim VALUES (
    'Long Event',
    '> 6 sessions'
);

SELECT
    *
FROM
    programlengthdim;
```

```
--EventSizeDIM
DROP TABLE eventsizedim;

CREATE TABLE eventsizedim (
   size_id      VARCHAR(20),
   description  VARCHAR(20)
);

INSERT INTO eventsizedim VALUES (
   'Small Event',
   '<=10'
);

INSERT INTO eventsizedim VALUES (
   'Medium Event',
   '11-30'
);

INSERT INTO eventsizedim VALUES (
   'Large Event',
   '>30'
);

SELECT
   *
FROM
   eventsizedim;

--ProgramDIM

DROP TABLE programdim;

CREATE TABLE programdim
   AS
      SELECT
         program_id,
         program_name,
         program_details,
         program_fee,
         program_length,
         program_frequency
      FROM
         program;

SELECT
   *
FROM
   programdim;
```

```
--SubscribeTimeDIM
DROP TABLE stimedim;

CREATE TABLE stimedim
   AS
      SELECT DISTINCT
         subscription_date AS s_date
      FROM
         subscription;

DROP TABLE subscribetimedim;

CREATE TABLE subscribetimedim
   AS
      SELECT DISTINCT
         to_char(s_date, 'YYYYMM')      AS subscribetime_id,
         to_char(s_date, 'MM')          AS month,
         to_char(s_date, 'YYYY')        AS year
      FROM
         stimedim;

SELECT
   *
FROM
   subscribetimedim;

--AttendanceTimeDIM
DROP TABLE atimddim;

CREATE TABLE atimedim
   AS
      SELECT DISTINCT
         att_date AS a_date
      FROM
         attendance;

DROP TABLE attendancetimedim;

CREATE TABLE attendancetimedim
   AS
      SELECT DISTINCT
         to_char(a_date, 'YYYYMM')      AS attendancetime_id,
         to_char(a_date, 'MM')          AS month,
         to_char(a_date, 'YYYY')        AS year
      FROM
         atimedim;

SELECT
   *
FROM
```

```
    attendancetimedim;

--RegisterTimeDIM
DROP TABLE rtimddim;

CREATE TABLE rtimedim
    AS
        SELECT DISTINCT
            reg_date AS r_date
        FROM
            registration;

DROP TABLE registertimedim;

CREATE TABLE registertimedim
    AS
        SELECT DISTINCT
            to_char(r_date, 'YYYYMM')      AS registertime_id,
            to_char(r_date, 'MM')          AS month,
            to_char(r_date, 'YYYY')        AS year
        FROM
            rtimedim;

SELECT
    *
FROM
    registertimedim;

--InterestFact
DROP TABLE interest_temp;

CREATE TABLE interest_temp
    AS
        SELECT
            p.person_marital_status,
            i.topic_id,
            p.person_age,
            a.address_state,
            p.person_job,
            p.person_id
        FROM
            person          p,
            address         a,
            person_interest  i
        WHERE
                p.person_id = i.person_id
            AND a.address_id = p.address_id;

ALTER TABLE interest_temp ADD (
    age_id  VARCHAR(20),
   job     VARCHAR(20)
);

UPDATE interest_temp
SET
   job = 'Student'
WHERE
   person_job = 'Student';

UPDATE interest_temp
SET
   job = 'Staff'
WHERE
   person_job = 'Staff';

UPDATE interest_temp
SET
   job = 'Community'
WHERE
   job IS NULL;

UPDATE interest_temp
SET
   age_id = 'Child'
WHERE
      person_age >= 0
   AND person_age <= 16;

UPDATE interest_temp
SET
   age_id = 'Young Adults'
WHERE
      person_age >= 17
   AND person_age <= 30;

UPDATE interest_temp
SET
   age_id = 'Middle-aged Adults'
WHERE
      person_age >= 31
   AND person_age <= 45;

UPDATE interest_temp
SET
   age_id = 'Old-aged Adults'
WHERE
   person_age > 45;

SELECT
   *
```

```
FROM
    interest_temp;

DROP TABLE interestfact;

CREATE TABLE interestfact
    AS
        SELECT
            person_marital_status,
            topic_id,
            age_id,
            address_state,
            job,
            COUNT(*) AS numofpeopleinterested
        FROM
            interest_temp
        GROUP BY
            person_marital_status,
            topic_id,
            age_id,
            address_state,
            job;
SELECT
    *
FROM
    interestfact;
--SubscriptionFact
DROP TABLE subscription_temp;

CREATE TABLE subscription_temp
    AS
        SELECT
            pr.program_id,
            to_char(s.subscription_date, 'YYYYMM') AS subscribetime_id,
            p.person_age,
            a.address_state,
            p.person_marital_status,
            p.person_job,
            s.person_id,
            pr.program_length
        FROM
            person       p,
            address      a,
            subscription  s,
            program      pr
        WHERE
                p.person_id = s.person_id
            AND a.address_id = p.address_id
            AND s.program_id = pr.program_id;
```

```
ALTER TABLE subscription_temp ADD (
   length_id  VARCHAR(20),
   job        VARCHAR(20),
   age_id     VARCHAR(50)
);

SELECT
   *
FROM
   subscription_temp;

UPDATE subscription_temp
SET
   age_id = 'Child'
WHERE
      person_age >= 0
   AND person_age <= 16;

UPDATE subscription_temp
SET
   age_id = 'Young Adults'
WHERE
      person_age >= 17
   AND person_age <= 30;

UPDATE subscription_temp
SET
   age_id = 'Middle-aged Adults'
WHERE
      person_age >= 31
   AND person_age <= 45;

UPDATE subscription_temp
SET
   age_id = 'Old-aged Adults'
WHERE
   person_age > 45;

UPDATE subscription_temp
SET
   job = 'Student'
WHERE
   person_job = 'Student';

UPDATE subscription_temp
SET
   job = 'Staff'
WHERE
   person_job = 'Staff';
```

```
UPDATE subscription_temp
SET
   job = 'Community'
WHERE
   job IS NULL;

UPDATE subscription_temp
SET
   program_length = substr(program_length, 0, instr(program_length, ' ') - 1);

UPDATE subscription_temp
SET
   length_id = 'Short Event'
WHERE
   program_length < 3;

UPDATE subscription_temp
SET
   length_id = 'Medium Event'
WHERE
      program_length > 3
   AND program_length < 6;

UPDATE subscription_temp
SET
   length_id = 'Long Event'
WHERE
   program_length > 6;

DROP TABLE subscriptionfact;

CREATE TABLE subscriptionfact
   AS
      SELECT
         program_id,
         subscribetime_id,
         age_id,
         address_state,
         person_marital_status,
         job,
         length_id,
         COUNT(DISTINCT person_id) AS numberofpeoplesubscribed
      FROM
         subscription_temp
      GROUP BY
         program_id,
         subscribetime_id,
         age_id,
         address_state,
         person_marital_status,
         job,
         length_id;
SELECT
   *
FROM
   subscriptionfact;
--RegistrationFact
DROP TABLE scd1_registration;

CREATE TABLE scd1_registration
   AS
      SELECT
         reg_id,
         reg_num_of_people_registered,
         reg_date,
         event_id,
         person_id,
         media_id
      FROM
         (
            SELECT
               reg_id,
               reg_num_of_people_registered,
               reg_date,
               event_id,
               person_id,
               media_id,
               RANK()
               OVER(PARTITION BY event_id, person_id
                   ORDER BY reg_id DESC
               ) AS rank
            FROM
               registration
         ) r
      WHERE
         r.rank = 1; /*if same person register the same event multiple times, we treat the
newest registration as the real registration*/ DROP TABLE register_temp;

CREATE TABLE register_temp
   AS
      SELECT
         r.media_id,
         to_char(r.reg_date, 'YYYYMM') AS registertime_id,
         p.person_job,
         e.event_size,
         r.reg_num_of_people_registered
      FROM
         scd1_registration  r,
         person             p,
         event              e
```

```
        WHERE
                r.person_id = p.person_id
            AND e.event_id = r.event_id;

ALTER TABLE register_temp ADD (
    size_id  VARCHAR(20),
    job      VARCHAR(20)
);

UPDATE register_temp
SET
    size_id = 'Small Event'
WHERE
    event_size <= 10;

UPDATE register_temp
SET
    size_id = 'Medium Event'
WHERE
        event_size > 10
    AND event_size <= 30;

UPDATE register_temp
SET
    size_id = 'Large Event'
WHERE
    event_size > 30;

UPDATE register_temp
SET
    job = 'Student'
WHERE
    person_job = 'Student';

UPDATE register_temp
SET
    job = 'Staff'
WHERE
    person_job = 'Staff';

UPDATE register_temp
SET
    job = 'Community'
WHERE
    job IS NULL;

DROP TABLE registerfact;

CREATE TABLE registerfact
    AS
```

```
        SELECT
            media_id,
            registertime_id,
            job,
            size_id,
            SUM(reg_num_of_people_registered) AS numofpeopleregistered
        FROM
            register_temp
        GROUP BY
            media_id,
            registertime_id,
            job,
            size_id;

SELECT
    *
FROM
    registerfact;

--AttendanceFact
DROP TABLE attendance_temp;

CREATE TABLE attendance_temp
    AS
        SELECT
            pr.program_id,
            to_char(att.att_date, 'YYYYMM') AS attendancetime_id,
            e.event_size,
            pr.program_length,
            p.person_job,
            ad.address_state,
            att.att_num_of_people_attended,
            att.att_donation_amount
        FROM
            program     pr,
            event       e,
            attendance  att,
            person      p,
            address     ad
        WHERE
                pr.program_id = e.program_id
            AND e.event_id = att.event_id
            AND p.person_id = att.person_id
            AND ad.address_id = p.address_id;

SELECT
    *
FROM
    attendance_temp;
```

```
ALTER TABLE attendance_temp ADD (
   size_id    VARCHAR(20),
   length_id  VARCHAR(20),
   job        VARCHAR(20)
);

UPDATE attendance_temp
SET
   size_id = 'Small Event'
WHERE
   event_size <= 10;

UPDATE attendance_temp
SET
   size_id = 'Medium Event'
WHERE
      event_size > 10
   AND event_size <= 30;

UPDATE attendance_temp
SET
   size_id = 'Large Event'
WHERE
   event_size > 30;

UPDATE attendance_temp
SET
   program_length = substr(program_length, 0, instr(program_length, ' ') - 1);

UPDATE attendance_temp
SET
   length_id = 'Short Event'
WHERE
   program_length < 3;

UPDATE attendance_temp
SET
   length_id = 'Medium Event'
WHERE
      program_length > 3
   AND program_length < 6;

UPDATE attendance_temp
SET
   length_id = 'Long Event'
WHERE
   program_length > 6;

UPDATE attendance_temp
SET
```

```
   job = 'Student'
WHERE
   person_job = 'Student';

UPDATE attendance_temp
SET
   job = 'Staff'
WHERE
   person_job = 'Staff';

UPDATE attendance_temp
SET
   job = 'Community'
WHERE
   job IS NULL;

SELECT
   *
FROM
   attendance_temp;

DROP TABLE attendancefact;

CREATE TABLE attendancefact
   AS
      SELECT
         program_id,
         attendancetime_id,
         size_id,
         length_id,
         job,
         address_state,
         SUM(att_num_of_people_attended)  AS numofpeopleattended,
         SUM(att_donation_amount)         AS totaldonation
      FROM
         attendance_temp
      GROUP BY
         program_id,
         attendancetime_id,
         size_id,
         length_id,
         job,
         address_state;

SELECT
   *
FROM
   attendancefact;
```

b) SQL statements Version-2

```
---------------------------------------------------------------------------
--construct star/snowflake schema
---------------------------------------------------------------------------
--Version-2 No aggregation (Level 0)
---dimensions

--persondim
CREATE TABLE persondim
   AS
      SELECT DISTINCT
         person_id
      FROM
         person;

SELECT
   *
FROM
   persondim;

--subscribeDayTimeDIM
DROP TABLE subscribedaytimedim;

CREATE TABLE subscribedaytimedim
   AS
      SELECT DISTINCT
         to_char(subscription_date, 'YYYYMMDD')  AS subscribe_id,
         to_char(subscription_date, 'MM')        AS month,
         to_char(subscription_date, 'YYYY')      AS year,
         to_char(subscription_date, 'DD')        AS day
      FROM
         subscription;

SELECT
   *
FROM
   subscribedaytimedim;
 --registerDayTimeDIM
DROP TABLE registerdaytimedim;

CREATE TABLE registerdaytimedim
   AS
      SELECT DISTINCT
         to_char(reg_date, 'YYYYMMDD')  AS registertime_id,
         to_char(reg_date, 'MM')        AS month,
         to_char(reg_date, 'YYYY')      AS year,
         to_char(reg_date, 'DD')        AS day
      FROM
```

```
            registration;

SELECT
    *
FROM
    registerdaytimedim;

--attendanceDayTimeDIM
DROP TABLE attendancedaytimedim;

CREATE TABLE attendancedaytimedim
    AS
        SELECT DISTINCT
            to_char(att_date, 'YYYYMMDD')  AS attendancetime_id,
            to_char(att_date, 'MM')        AS month,
            to_char(att_date, 'YYYY')      AS year,
            to_char(att_date, 'DD')        AS day
        FROM
            attendance;

SELECT
    *
FROM
    attendancedaytimedim;

-- eventdim
CREATE TABLE eventdim
    AS
        SELECT DISTINCT
            event_id
        FROM
            event;

SELECT
    *
FROM
    eventdim;

--attendtypedim
CREATE TABLE attendtypedim (
    attendedtimes VARCHAR(30)
);

INSERT INTO attendtypedim VALUES ( 'first time' );

INSERT INTO attendtypedim VALUES ( 'not first time' );

SELECT
    *
FROM
```

```
   attendtypedim;

--InterestFact0

SELECT
   *
FROM
   interest_temp;

CREATE TABLE interestfact0
   AS
      SELECT
         person_id,
         person_marital_status,
         topic_id,
         age_id,
         address_state,
         job,
         COUNT(*) AS numofpeopleinterested
      FROM
         interest_temp
      GROUP BY
         person_id,
         person_marital_status,
         topic_id,
         age_id,
         address_state,
         job;

SELECT
   *
FROM
   interestfact0;

--subscriptionfact0
DROP TABLE subscription_temp0;

CREATE TABLE subscription_temp0
   AS
      SELECT
         pr.program_id,
         to_char(s.subscription_date, 'YYYYMMDD') AS subscribetime_id,
         p.person_age,
         a.address_state,
         p.person_marital_status,
         p.person_job,
         s.person_id,
         pr.program_length
      FROM
         person     p,
            address      a,
            subscription  s,
            program      pr
        WHERE
                p.person_id = s.person_id
            AND a.address_id = p.address_id
            AND s.program_id = pr.program_id;

ALTER TABLE subscription_temp0 ADD (
    length_id  VARCHAR(20),
    job        VARCHAR(20),
    age_id     VARCHAR(50)
);

SELECT
    *
FROM
    subscription_temp0;

UPDATE subscription_temp0
SET
    age_id = 'Child'
WHERE
        person_age >= 0
    AND person_age <= 16;

UPDATE subscription_temp0
SET
    age_id = 'Young Adults'
WHERE
        person_age >= 17
    AND person_age <= 30;

UPDATE subscription_temp0
SET
    age_id = 'Middle-aged Adults'
WHERE
        person_age >= 31
    AND person_age <= 45;

UPDATE subscription_temp0
SET
    age_id = 'Old-aged Adults'
WHERE
    person_age > 45;

UPDATE subscription_temp0
SET
    job = 'Student'
WHERE
```

```
   person_job = 'Student';

UPDATE subscription_temp0
SET
   job = 'Staff'
WHERE
   person_job = 'Staff';

UPDATE subscription_temp0
SET
   job = 'Community'
WHERE
   job IS NULL;

UPDATE subscription_temp0
SET
   program_length = substr(program_length, 0, instr(program_length, ' ') - 1);

UPDATE subscription_temp0
SET
   length_id = 'Short Event'
WHERE
   program_length < 3;

UPDATE subscription_temp0
SET
   length_id = 'Medium Event'
WHERE
      program_length > 3
   AND program_length < 6;

UPDATE subscription_temp0
SET
   length_id = 'Long Event'
WHERE
   program_length > 6;

DROP TABLE subscriptionfact0;

CREATE TABLE subscriptionfact0
   AS
      SELECT
         person_id,
         program_id,
         subscribetime_id,
         age_id,
         address_state,
         person_marital_status,
         job,
         length_id,
         COUNT(DISTINCT person_id) AS numberofpeoplesubscribed
      FROM
         subscription_temp0
      GROUP BY
         person_id,
         program_id,
         subscribetime_id,
         age_id,
         address_state,
         person_marital_status,
         job,
         length_id;

SELECT
   *
FROM
   subscriptionfact0;

SELECT
   *
FROM
   subscriptionfact0
WHERE
   numberofpeoplesubscribed > 1;

--registerfact0
DROP TABLE scd1_registration0;

CREATE TABLE scd1_registration0
   AS
      SELECT
         reg_id,
         reg_num_of_people_registered,
         reg_date,
         event_id,
         person_id,
         media_id
      FROM
         (
             SELECT
                 reg_id,
                 reg_num_of_people_registered,
                 reg_date,
                 event_id,
                 person_id,
                 media_id,
                 RANK()
                 OVER(PARTITION BY event_id, person_id
                      ORDER BY reg_id DESC
                 ) AS rank
            FROM
               registration
        ) r
      WHERE
         r.rank = 1; /*if same person register the same event multiple times, we treat the
newest registration as the real registration*/ DROP TABLE register_temp0;

CREATE TABLE register_temp0
   AS
      SELECT
         p.person_id,
         e.event_id,
         r.media_id,
         to_char(r.reg_date, 'YYYYMMDD') AS registertime_id,
         p.person_job,
         e.event_size,
         r.reg_num_of_people_registered
      FROM
         scd1_registration0  r,
         person             p,
         event              e
      WHERE
            r.person_id = p.person_id
         AND e.event_id = r.event_id;

ALTER TABLE register_temp0 ADD (
   size_id  VARCHAR(20),
   job      VARCHAR(20)
);

UPDATE register_temp0
SET
   size_id = 'Small Event'
WHERE
   event_size <= 10;

UPDATE register_temp0
SET
   size_id = 'Medium Event'
WHERE
      event_size > 10
   AND event_size <= 30;

UPDATE register_temp0
SET
   size_id = 'Large Event'
WHERE
   event_size > 30;

UPDATE register_temp0
```

```
SET
   job = 'Student'
WHERE
   person_job = 'Student';

UPDATE register_temp0
SET
   job = 'Staff'
WHERE
   person_job = 'Staff';

UPDATE register_temp0
SET
   job = 'Community'
WHERE
   job IS NULL;

DROP TABLE registerfact0;

CREATE TABLE registerfact0
   AS
      SELECT
         person_id,
         event_id,
         media_id,
         registertime_id,
         job,
         size_id,
         SUM(reg_num_of_people_registered) AS numofpeopleregistered
      FROM
         register_temp0
      GROUP BY
         person_id,
         event_id,
         media_id,
         registertime_id,
         job,
         size_id;

SELECT
   *
FROM
   registerfact0;

SELECT
   COUNT(*)
FROM
   registerfact0; /*1412*/ SELECT
   COUNT(*)
FROM
```

```
   scd1_registration0; /*1412 two values are the same each which means no aggregation
happened */

--attendancefact0
DROP TABLE alterattendance;

CREATE TABLE alterattendance
   AS
      SELECT
         att_id,
         att_date,
         att_donation_amount,
         att_num_of_people_attended,
         event_id,
         person_id,
         rank
      FROM
         (
            SELECT
               att_id,
               att_date,
               att_donation_amount,
               att_num_of_people_attended,
               event_id,
               person_id,
               RANK()
               OVER(PARTITION BY event_id, person_id, att_date
                  ORDER BY att_id
               ) AS rank
            FROM
               attendance
         );

SELECT
   *
FROM
   alterattendance
WHERE
      event_id = 150
   AND person_id = 'PE005'
   AND att_date = TO_DATE('2020-09-10', 'YYYY-MM-DD');

SELECT
   *
FROM
   alterattendance;

DROP TABLE attendance_temp0;

CREATE TABLE attendance_temp0
```

```
  AS
    SELECT
      p.person_id,
      e.event_id,
      pr.program_id,
      to_char(att.att_date, 'YYYYMMDD') AS attendancetime_id,
      e.event_size,
      pr.program_length,
      p.person_job,
      ad.address_state,
      att.att_num_of_people_attended,
      att.att_donation_amount,
      att.rank
    FROM
      program          pr,
      event            e,
      alterattendance  att,
      person           p,
      address          ad
    WHERE
        pr.program_id = e.program_id
      AND e.event_id = att.event_id
      AND p.person_id = att.person_id
      AND ad.address_id = p.address_id;

SELECT
  *
FROM
  attendance_temp0;

ALTER TABLE attendance_temp0 ADD (
  size_id        VARCHAR(20),
  length_id      VARCHAR(20),
  job            VARCHAR(20),
  attendedtimes  VARCHAR(20)
);

UPDATE attendance_temp0
SET
  attendedtimes = 'first time'
WHERE
  rank = 1;

UPDATE attendance_temp0
SET
  attendedtimes = 'not first time'
WHERE
  rank <> 1;

UPDATE attendance_temp0
```

```
SET
   size_id = 'Small Event'
WHERE
   event_size <= 10;

UPDATE attendance_temp0
SET
   size_id = 'Medium Event'
WHERE
      event_size > 10
   AND event_size <= 30;

UPDATE attendance_temp0
SET
   size_id = 'Large Event'
WHERE
   event_size > 30;

UPDATE attendance_temp0
SET
   program_length = substr(program_length, 0, instr(program_length, ' ') - 1);

UPDATE attendance_temp0
SET
   length_id = 'Short Event'
WHERE
   program_length < 3;

UPDATE attendance_temp0
SET
   length_id = 'Medium Event'
WHERE
      program_length > 3
   AND program_length < 6;

UPDATE attendance_temp0
SET
   length_id = 'Long Event'
WHERE
   program_length > 6;

UPDATE attendance_temp0
SET
   job = 'Student'
WHERE
   person_job = 'Student';

UPDATE attendance_temp0
SET
   job = 'Staff'
```

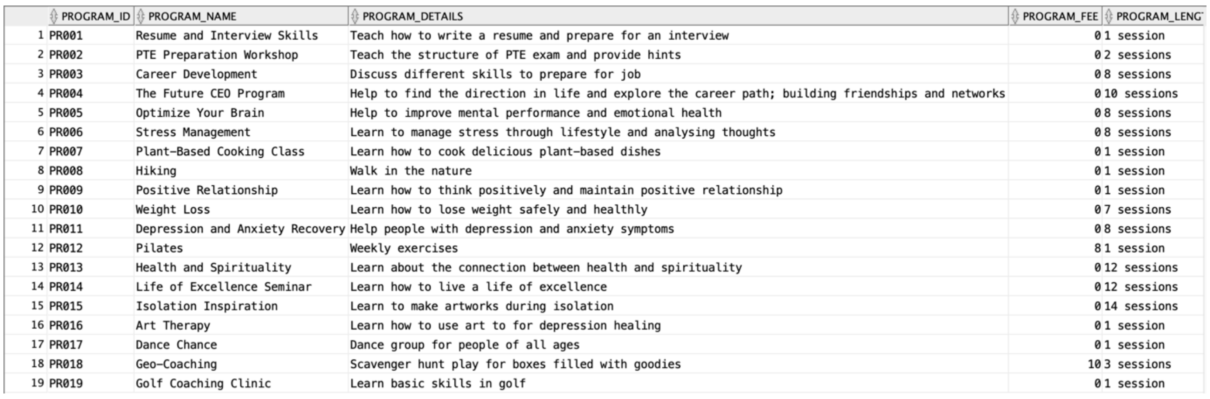
```
WHERE
   person_job = 'Staff';

UPDATE attendance_temp0
SET
   job = 'Community'
WHERE
   job IS NULL;

SELECT
   *
FROM
   attendance_temp0;

DROP TABLE attendancefact0;

CREATE TABLE attendancefact0
   AS
      SELECT
         person_id,
         event_id,
         attendedtimes,
         program_id,
         attendancetime_id,
         size_id,
         length_id,
         job,
         address_state,
         SUM(att_num_of_people_attended)  AS numofpeopleattended,
         SUM(att_donation_amount)         AS totaldonation
      FROM
         attendance_temp0
      GROUP BY
         person_id,
         event_id,
         attendedtimes,
         program_id,
         attendancetime_id,
         size_id,
         length_id,
         job,
         address_state;

SELECT
   *
FROM
   attendancefact0;
---testtable---
/*create table testtable as
```

```
select att.att_id, p.person_id, e.event_id,pr.program_id, to_char(att.att_date, 'YYYYMMDD')
as attendanceTime_id,
e.event_size, pr.program_length, p.person_job, ad.address_state,
att.att_num_of_people_attended, att.att_donation_amount, att.rank
from program pr, event e, alterattendance att, person p, address ad
where pr.program_id = e.program_id
and e.event_id = att.event_id
and p.person_id = att.person_id
and ad.address_id = p.address_id;*/
--select * from attendance where att_id not in (select att_id from testtable);
SELECT
   COUNT(*)
FROM
   attendance; /*5650*/ SELECT
   COUNT(*)
FROM
   attendancefact0; /*5650, the fact table have the same number of rows as the original
attendance table. so there is no aggregation happened*/ SELECT
   *
FROM
   attendancefact0;
```

c) Version-1 Screenshots

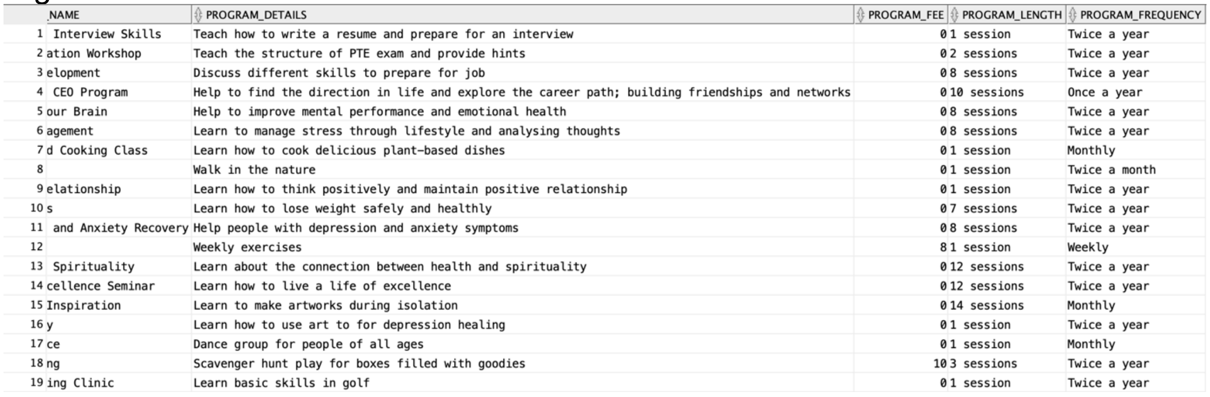
ProgramDIM

Left-hand-side columns



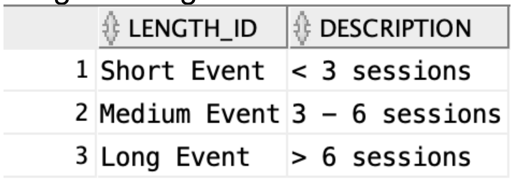
	PROGRAM_ID	PROGRAM_NAME	PROGRAM_DETAILS	PROGRAM_FEE	PROGRAM_LENG
1	PR001	Resume and Interview Skills	Teach how to write a resume and prepare for an interview	0	1 session
2	PR002	PTE Preparation Workshop	Teach the structure of PTE exam and provide hints	0	2 sessions
3	PR003	Career Development	Discuss different skills to prepare for job	0	8 sessions
4	PR004	The Future CEO Program	Help to find the direction in life and explore the career path; building friendships and networks	0	10 sessions
5	PR005	Optimize Your Brain	Help to improve mental performance and emotional health	0	8 sessions
6	PR006	Stress Management	Learn to manage stress through lifestyle and analysing thoughts	0	8 sessions
7	PR007	Plant-Based Cooking Class	Learn how to cook delicious plant-based dishes	0	1 session
8	PR008	Hiking	Walk in the nature	0	1 session
9	PR009	Positive Relationship	Learn how to think positively and maintain positive relationship	0	1 session
10	PR010	Weight Loss	Learn how to lose weight safely and healthly	0	7 sessions
11	PR011	Depression and Anxiety Recovery	Help people with depression and anxiety symptoms	0	8 sessions
12	PR012	Pilates	Weekly exercises	8	1 session
13	PR013	Health and Spirituality	Learn about the connection between health and spirituality	0	12 sessions
14	PR014	Life of Excellence Seminar	Learn how to live a life of excellence	0	12 sessions
15	PR015	Isolation Inspiration	Learn to make artworks during isolation	0	14 sessions
16	PR016	Art Therapy	Learn how to use art to for depression healing	0	1 session
17	PR017	Dance Chance	Dance group for people of all ages	0	1 session
18	PR018	Geo-Coaching	Scavenger hunt play for boxes filled with goodies	10	3 sessions
19	PR019	Golf Coaching Clinic	Learn basic skills in golf	0	1 session

Right hand side columns



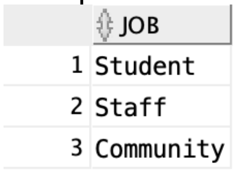
	NAME	PROGRAM_DETAILS	PROGRAM_FEE	PROGRAM_LENGTH	PROGRAM_FREQUENCY
1	Interview Skills	Teach how to write a resume and prepare for an interview	0	1 session	Twice a year
2	ation Workshop	Teach the structure of PTE exam and provide hints	0	2 sessions	Twice a year
3	elopment	Discuss different skills to prepare for job	0	8 sessions	Twice a year
4	CEO Program	Help to find the direction in life and explore the career path; building friendships and networks	0	10 sessions	Once a year
5	our Brain	Help to improve mental performance and emotional health	0	8 sessions	Twice a year
6	agement	Learn to manage stress through lifestyle and analysing thoughts	0	8 sessions	Twice a year
7	d Cooking Class	Learn how to cook delicious plant-based dishes	0	1 session	Monthly
8		Walk in the nature	0	1 session	Twice a month
9	elationship	Learn how to think positively and maintain positive relationship	0	1 session	Twice a year
10	s	Learn how to lose weight safely and healthly	0	7 sessions	Twice a year
11	and Anxiety Recovery	Help people with depression and anxiety symptoms	0	8 sessions	Twice a year
12		Weekly exercises	8	1 session	Weekly
13	Spirituality	Learn about the connection between health and spirituality	0	12 sessions	Twice a year
14	cellence Seminar	Learn how to live a life of excellence	0	12 sessions	Twice a year
15	Inspiration	Learn to make artworks during isolation	0	14 sessions	Monthly
16	y	Learn how to use art to for depression healing	0	1 session	Twice a year
17	ce	Dance group for people of all ages	0	1 session	Monthly
18	ng	Scavenger hunt play for boxes filled with goodies	10	3 sessions	Twice a year
19	ing Clinic	Learn basic skills in golf	0	1 session	Twice a year

ProgramLengthDIM



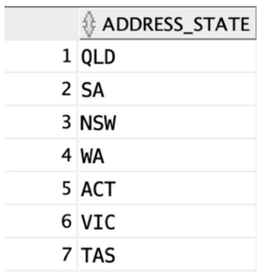
	LENGTH_ID	DESCRIPTION
1	Short Event	< 3 sessions
2	Medium Event	3 - 6 sessions
3	Long Event	> 6 sessions

OccupationDIM



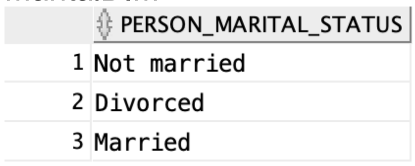
	JOB
1	Student
2	Staff
3	Community

StateDIM



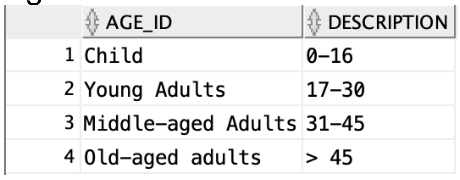
	ADDRESS_STATE
1	QLD
2	SA
3	NSW
4	WA
5	ACT
6	VIC
7	TAS

MaritalDIM



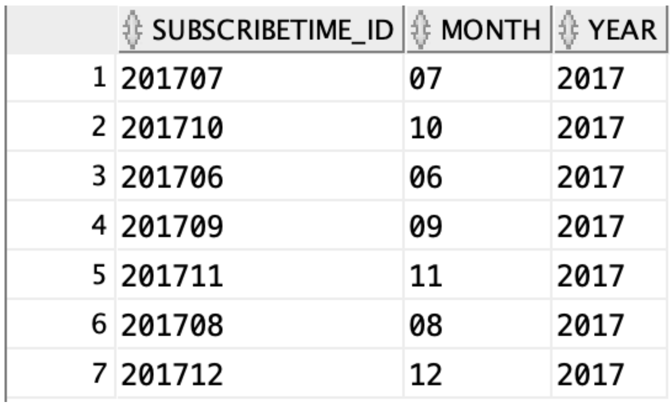
	PERSON_MARITAL_STATUS
1	Not married
2	Divorced
3	Married

AgeDIM



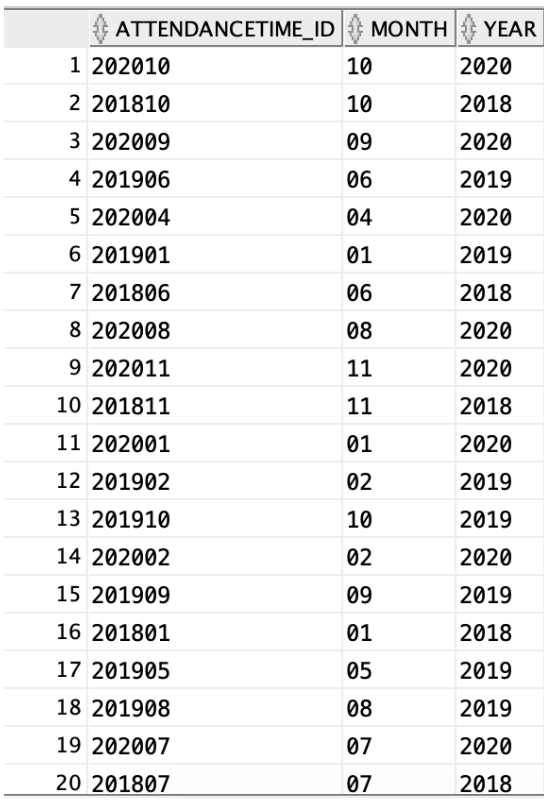
	AGE_ID	DESCRIPTION
1	Child	0-16
2	Young Adults	17-30
3	Middle-aged Adults	31-45
4	Old-aged adults	> 45

SubscribeTimeDIM



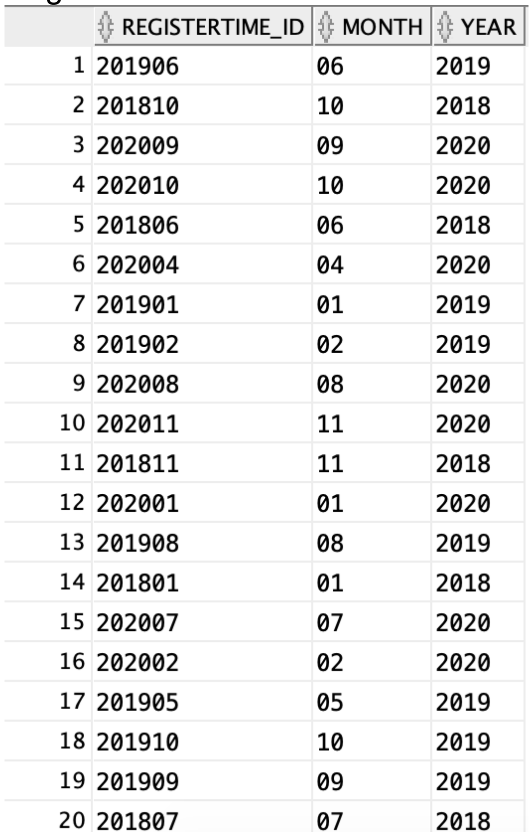
	SUBSCRIBETIME_ID	MONTH	YEAR
1	201707	07	2017
2	201710	10	2017
3	201706	06	2017
4	201709	09	2017
5	201711	11	2017
6	201708	08	2017
7	201712	12	2017

AttendanceTimeDIM



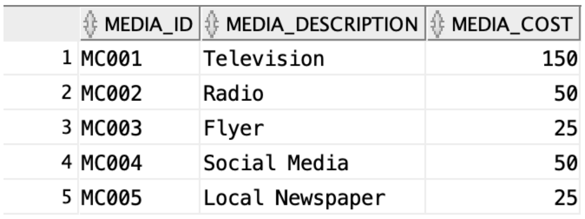
	ATTENDANCETIME_ID	MONTH	YEAR
1	202010	10	2020
2	201810	10	2018
3	202009	09	2020
4	201906	06	2019
5	202004	04	2020
6	201901	01	2019
7	201806	06	2018
8	202008	08	2020
9	202011	11	2020
10	201811	11	2018
11	202001	01	2020
12	201902	02	2019
13	201910	10	2019
14	202002	02	2020
15	201909	09	2019
16	201801	01	2018
17	201905	05	2019
18	201908	08	2019
19	202007	07	2020
20	201807	07	2018

RegisterTimeDIM



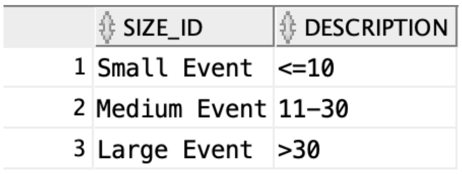
	REGISTERTIME_ID	MONTH	YEAR
1	201906	06	2019
2	201810	10	2018
3	202009	09	2020
4	202010	10	2020
5	201806	06	2018
6	202004	04	2020
7	201901	01	2019
8	201902	02	2019
9	202008	08	2020
10	202011	11	2020
11	201811	11	2018
12	202001	01	2020
13	201908	08	2019
14	201801	01	2018
15	202007	07	2020
16	202002	02	2020
17	201905	05	2019
18	201910	10	2019
19	201909	09	2019
20	201807	07	2018

MediaDIM



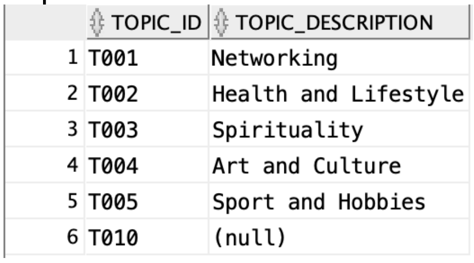
	MEDIA_ID	MEDIA_DESCRIPTION	MEDIA_COST
1	MC001	Television	150
2	MC002	Radio	50
3	MC003	Flyer	25
4	MC004	Social Media	50
5	MC005	Local Newspaper	25

EventSizeDIM



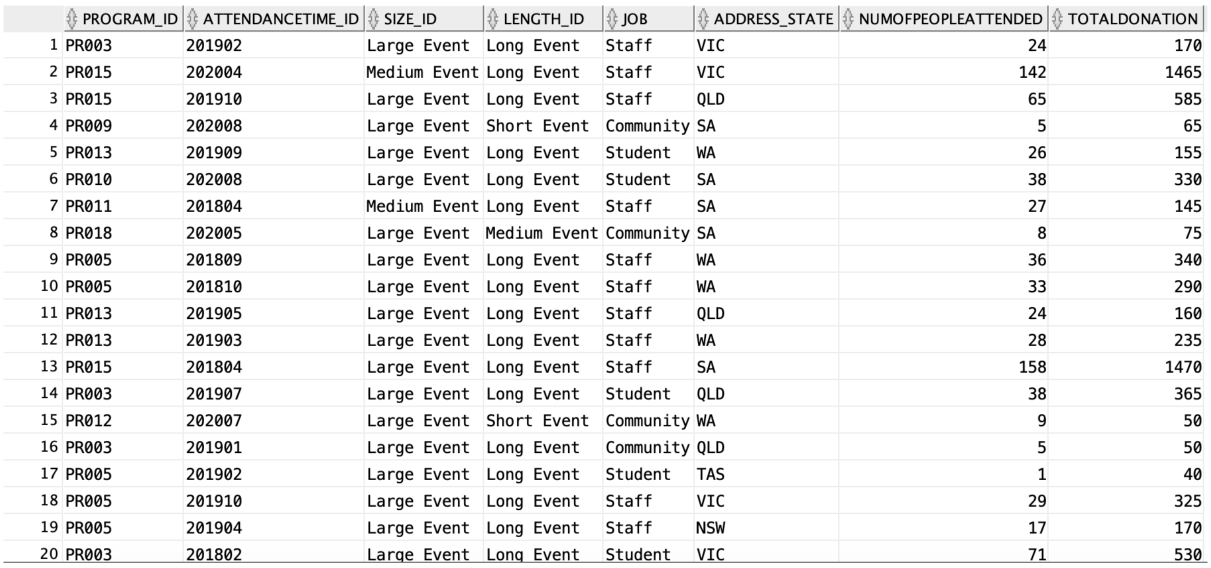
	SIZE_ID	DESCRIPTION
1	Small Event	<=10
2	Medium Event	11–30
3	Large Event	>30

TopicDIM



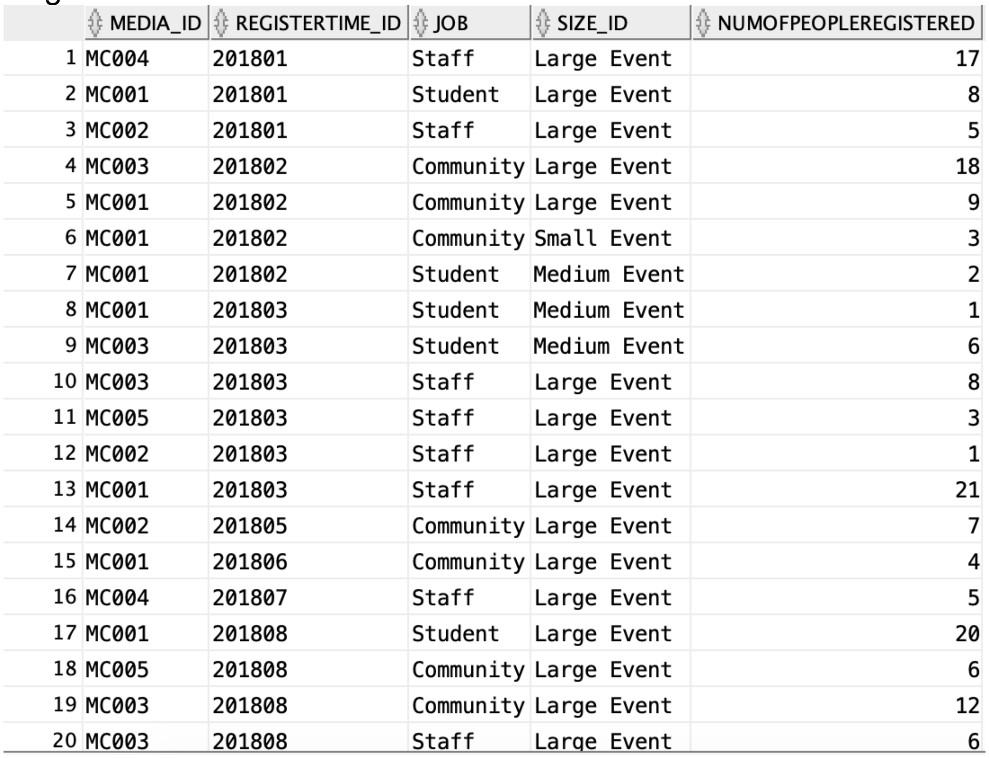
	TOPIC_ID	TOPIC_DESCRIPTION
1	T001	Networking
2	T002	Health and Lifestyle
3	T003	Spirituality
4	T004	Art and Culture
5	T005	Sport and Hobbies
6	T010	(null)

AttendanceFact



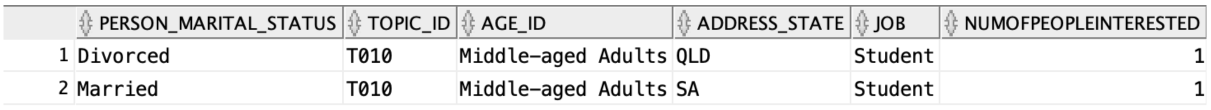
	PROGRAM_ID	ATTENDANCETIME_ID	SIZE_ID	LENGTH_ID	JOB	ADDRESS_STATE	NUMOFPEOPLEATTENDED	TOTALDONATION
1	PR003	201902	Large Event	Long Event	Staff	VIC	24	170
2	PR015	202004	Medium Event	Long Event	Staff	VIC	142	1465
3	PR015	201910	Large Event	Long Event	Staff	QLD	65	585
4	PR009	202008	Large Event	Short Event	Community	SA	5	65
5	PR013	201909	Large Event	Long Event	Student	WA	26	155
6	PR010	202008	Large Event	Long Event	Student	SA	38	330
7	PR011	201804	Medium Event	Long Event	Staff	SA	27	145
8	PR018	202005	Large Event	Medium Event	Community	SA	8	75
9	PR005	201809	Large Event	Long Event	Staff	WA	36	340
10	PR005	201810	Large Event	Long Event	Staff	WA	33	290
11	PR013	201905	Large Event	Long Event	Staff	QLD	24	160
12	PR013	201903	Large Event	Long Event	Staff	WA	28	235
13	PR015	201804	Large Event	Long Event	Staff	SA	158	1470
14	PR003	201907	Large Event	Long Event	Student	QLD	38	365
15	PR012	202007	Large Event	Short Event	Community	WA	9	50
16	PR003	201901	Large Event	Long Event	Community	QLD	5	50
17	PR005	201902	Large Event	Long Event	Student	TAS	1	40
18	PR005	201910	Large Event	Long Event	Staff	VIC	29	325
19	PR005	201904	Large Event	Long Event	Staff	NSW	17	170
20	PR003	201802	Large Event	Long Event	Student	VIC	71	530

RegisterFact



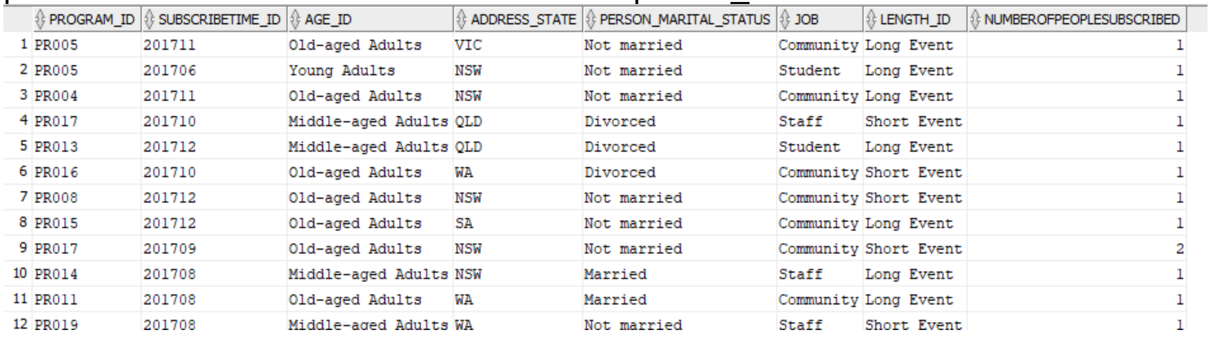
	MEDIA_ID	REGISTERTIME_ID	JOB	SIZE_ID	NUMOFPEOPLEREGISTERED
1	MC004	201801	Staff	Large Event	17
2	MC001	201801	Student	Large Event	8
3	MC002	201801	Staff	Large Event	5
4	MC003	201802	Community	Large Event	18
5	MC001	201802	Community	Large Event	9
6	MC001	201802	Community	Small Event	3
7	MC001	201802	Student	Medium Event	2
8	MC001	201803	Student	Medium Event	1
9	MC003	201803	Student	Medium Event	6
10	MC003	201803	Staff	Large Event	8
11	MC005	201803	Staff	Large Event	3
12	MC002	201803	Staff	Large Event	1
13	MC001	201803	Staff	Large Event	21
14	MC002	201805	Community	Large Event	7
15	MC001	201806	Community	Large Event	4
16	MC004	201807	Staff	Large Event	5
17	MC001	201808	Student	Large Event	20
18	MC005	201808	Community	Large Event	6
19	MC003	201808	Community	Large Event	12
20	MC003	201808	Staff	Large Event	6

InterestFact



	PERSON_MARITAL_STATUS	TOPIC_ID	AGE_ID	ADDRESS_STATE	JOB	NUMOFPEOPLEINTERESTED
1	Divorced	T010	Middle-aged Adults	QLD	Student	1
2	Married	T010	Middle-aged Adults	SA	Student	1

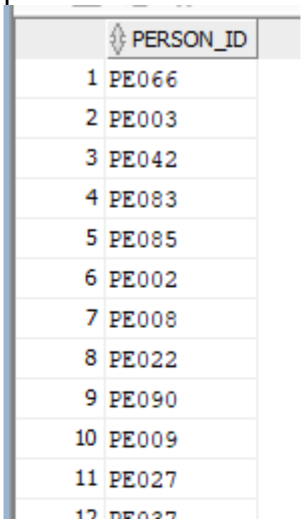
SubscriptionFact
assumption: a person can subscribe twice to the same event at the same time, but we will not count this as multiple number of persons subscribed. because we can't say one person subscribed to an event 1000 times means this event has 1000 persons subscribed. we will take the distinct person_id in this case.



	PROGRAM_ID	SUBSCRIBETIME_ID	AGE_ID	ADDRESS_STATE	PERSON_MARITAL_STATUS	JOB	LENGTH_ID	NUMBEROFPEOPLESUBSCRIBED
1	PR005	201711	Old-aged Adults	VIC	Not married	Community	Long Event	1
2	PR005	201706	Young Adults	NSW	Not married	Student	Long Event	1
3	PR004	201711	Old-aged Adults	NSW	Not married	Community	Long Event	1
4	PR017	201710	Middle-aged Adults	QLD	Divorced	Staff	Short Event	1
5	PR013	201712	Middle-aged Adults	QLD	Divorced	Student	Long Event	1
6	PR016	201710	Old-aged Adults	WA	Divorced	Community	Short Event	1
7	PR008	201712	Old-aged Adults	NSW	Not married	Community	Short Event	1
8	PR015	201712	Old-aged Adults	SA	Not married	Community	Long Event	1
9	PR017	201709	Old-aged Adults	NSW	Not married	Community	Short Event	2
10	PR014	201708	Middle-aged Adults	NSW	Married	Staff	Long Event	1
11	PR011	201708	Old-aged Adults	WA	Married	Community	Long Event	1
12	PR019	201708	Middle-aged Adults	WA	Not married	Staff	Short Event	1

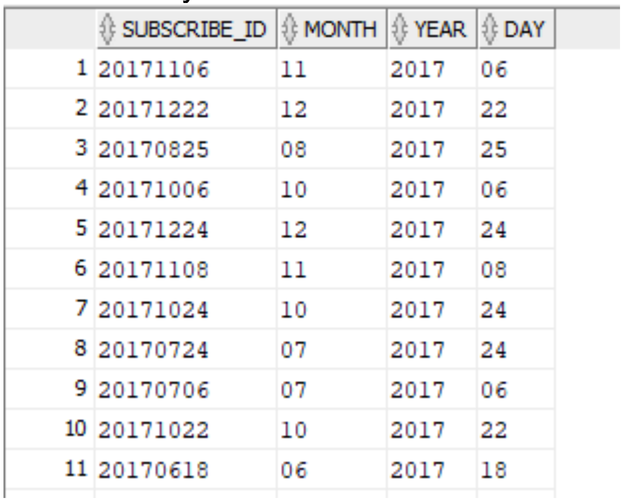
Version-2 Screenshots

personDIM



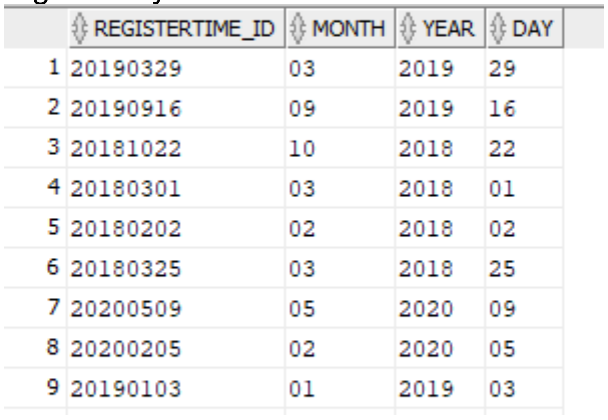
	PERSON_ID
1	PE066
2	PE003
3	PE042
4	PE083
5	PE085
6	PE002
7	PE008
8	PE022
9	PE090
10	PE009
11	PE027

subscribeDayTimeDIM



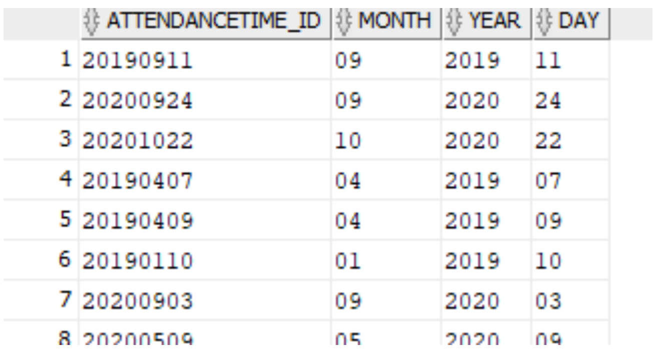
	SUBSCRIBE_ID	MONTH	YEAR	DAY
1	20171106	11	2017	06
2	20171222	12	2017	22
3	20170825	08	2017	25
4	20171006	10	2017	06
5	20171224	12	2017	24
6	20171108	11	2017	08
7	20171024	10	2017	24
8	20170724	07	2017	24
9	20170706	07	2017	06
10	20171022	10	2017	22
11	20170618	06	2017	18

registerDayTimeDIM



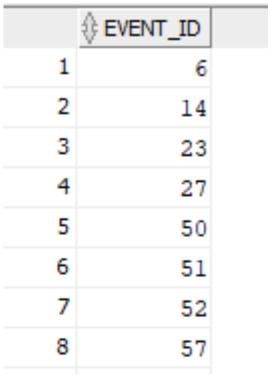
	REGISTERTIME_ID	MONTH	YEAR	DAY
1	20190329	03	2019	29
2	20190916	09	2019	16
3	20181022	10	2018	22
4	20180301	03	2018	01
5	20180202	02	2018	02
6	20180325	03	2018	25
7	20200509	05	2020	09
8	20200205	02	2020	05
9	20190103	01	2019	03

attendanceDayTimeDIM



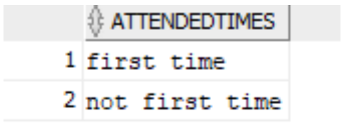
	ATTENDANCETIME_ID	MONTH	YEAR	DAY
1	20190911	09	2019	11
2	20200924	09	2020	24
3	20201022	10	2020	22
4	20190407	04	2019	07
5	20190409	04	2019	09
6	20190110	01	2019	10
7	20200903	09	2020	03
8	20200509	05	2020	09

eventdim



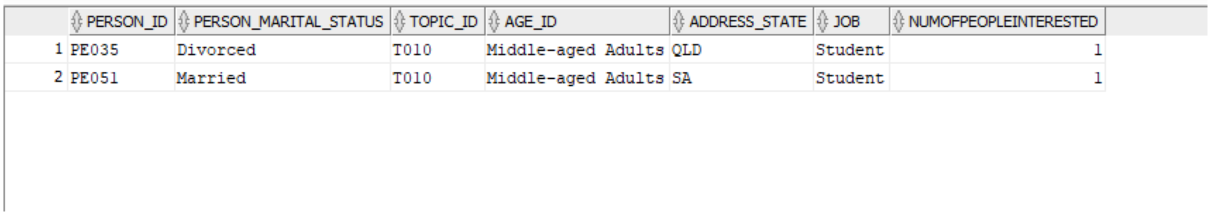
	EVENT_ID
1	6
2	14
3	23
4	27
5	50
6	51
7	52
8	57

attendtypedim
This is used to distinguish some people who attend the same event more than once in the same day. The assumption is that the person may attend in the beginning and invite his/her friend to this event later in the day.



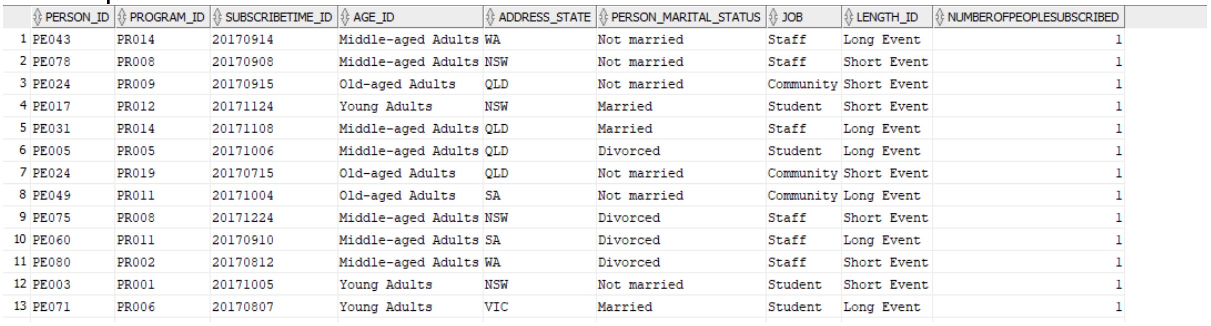
	ATTENDEDTIMES
1	first time
2	not first time

InterestFact0



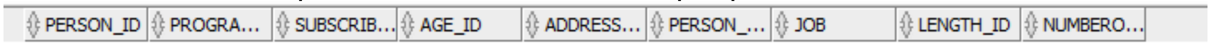
	PERSON_ID	PERSON_MARITAL_STATUS	TOPIC_ID	AGE_ID	ADDRESS_STATE	JOB	NUMOFPEOPLEINTERESTED
1	PE035	Divorced	T010	Middle-aged Adults	QLD	Student	1
2	PE051	Married	T010	Middle-aged Adults	SA	Student	1

subscriptionfact0



	PERSON_ID	PROGRAM_ID	SUBSCRIBETIME_ID	AGE_ID	ADDRESS_STATE	PERSON_MARITAL_STATUS	JOB	LENGTH_ID	NUMBEROFPEOPLESUBSCRIBED
1	PE043	PR014	20170914	Middle-aged Adults	WA	Not married	Staff	Long Event	1
2	PE078	PR008	20170908	Middle-aged Adults	NSW	Not married	Staff	Short Event	1
3	PE024	PR009	20170915	Old-aged Adults	QLD	Not married	Community	Short Event	1
4	PE017	PR012	20171124	Young Adults	NSW	Married	Student	Short Event	1
5	PE031	PR014	20171108	Middle-aged Adults	QLD	Married	Staff	Long Event	1
6	PE005	PR005	20171006	Middle-aged Adults	QLD	Divorced	Student	Long Event	1
7	PE024	PR019	20170715	Old-aged Adults	QLD	Not married	Community	Short Event	1
8	PE049	PR011	20171004	Old-aged Adults	SA	Not married	Community	Long Event	1
9	PE075	PR008	20171224	Middle-aged Adults	NSW	Divorced	Staff	Short Event	1
10	PE060	PR011	20170910	Middle-aged Adults	SA	Divorced	Staff	Long Event	1
11	PE080	PR002	20170812	Middle-aged Adults	WA	Divorced	Staff	Short Event	1
12	PE003	PR001	20171005	Young Adults	NSW	Not married	Student	Short Event	1
13	PE071	PR006	20170807	Young Adults	VIC	Married	Student	Long Event	1

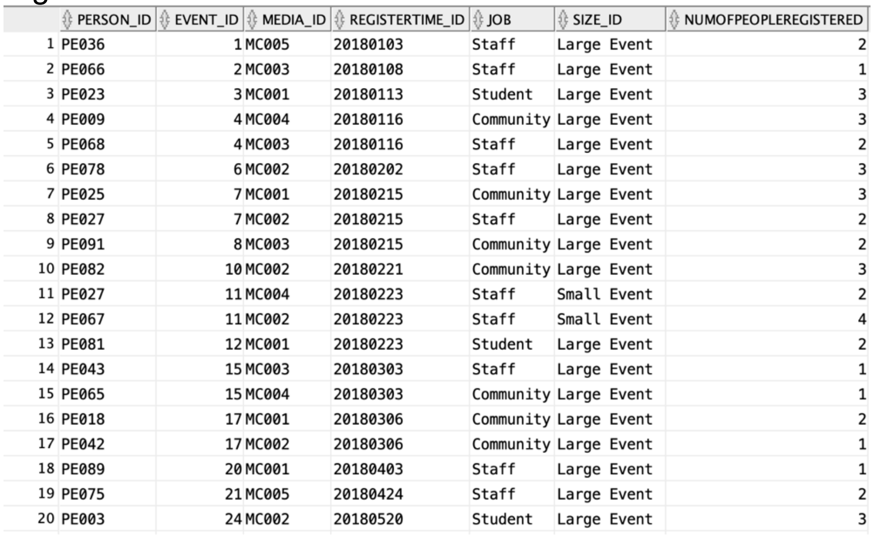
select * from Subscriptionfact0 where numberofpeoplesubscribed > 1;



PERSON_ID	PROGRA...	SUBSCRIB...	AGE_ID	ADDRESS...	PERSON_...	JOB	LENGTH_ID	NUMBERO...

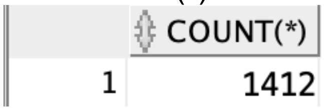
no number >1 means no aggregation here.

registerfact0



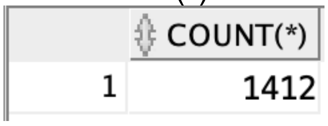
	PERSON_ID	EVENT_ID	MEDIA_ID	REGISTERTIME_ID	JOB	SIZE_ID	NUMOFPEOPLEREGISTERED
1	PE036	1	MC005	20180103	Staff	Large Event	2
2	PE066	2	MC003	20180108	Staff	Large Event	1
3	PE023	3	MC001	20180113	Student	Large Event	3
4	PE009	4	MC004	20180116	Community	Large Event	3
5	PE068	4	MC003	20180116	Staff	Large Event	2
6	PE078	6	MC002	20180202	Staff	Large Event	3
7	PE025	7	MC001	20180215	Community	Large Event	3
8	PE027	7	MC002	20180215	Staff	Large Event	2
9	PE091	8	MC003	20180215	Community	Large Event	2
10	PE082	10	MC002	20180221	Community	Large Event	3
11	PE027	11	MC004	20180223	Staff	Small Event	2
12	PE067	11	MC002	20180223	Staff	Small Event	4
13	PE081	12	MC001	20180223	Student	Large Event	2
14	PE043	15	MC003	20180303	Staff	Large Event	1
15	PE065	15	MC004	20180303	Community	Large Event	1
16	PE018	17	MC001	20180306	Community	Large Event	2
17	PE042	17	MC002	20180306	Community	Large Event	1
18	PE089	20	MC001	20180403	Staff	Large Event	1
19	PE075	21	MC005	20180424	Staff	Large Event	2
20	PE003	24	MC002	20180520	Student	Large Event	3

select count(*) from RegisterFact0;



	COUNT(*)
1	1412

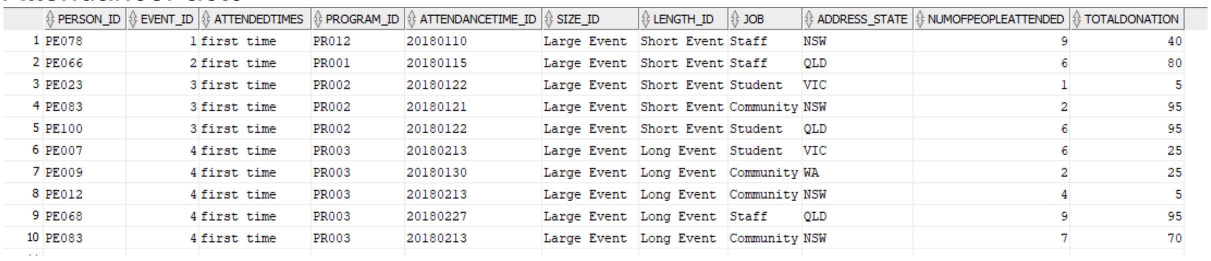
select count(*) from SCD1_registration0;



	COUNT(*)
1	1412

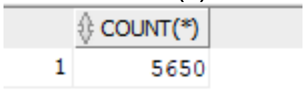
here we use the SCD1 to reflect the real registration value in the original ERD table, because some people may register multiple times for various reasons, for example, they want to change registration, they forgot they registered. So, we want to reflect the most recent change as the final one, so SCD1 will be very suitable here.
we can also see to count(*) are the same between the FACT and SCD1 process ERD table. We can say the numberofpeopleregistered values are not being aggregated and it remains the same as the one in original ERD table.

AttendanceFact0



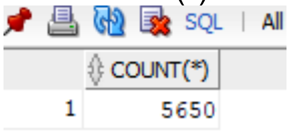
	PERSON_ID	EVENT_ID	ATTENDEDTIMES	PROGRAM_ID	ATTENDANCETIME_ID	SIZE_ID	LENGTH_ID	JOB	ADDRESS_STATE	NUMOFPEOPLEATTENDED	TOTALDONATION
1	PE078	1	first time	PR012	20180110	Large Event	Short Event	Staff	NSW	9	40
2	PE066	2	first time	PR001	20180115	Large Event	Short Event	Staff	QLD	6	80
3	PE023	3	first time	PR002	20180122	Large Event	Short Event	Student	VIC	1	5
4	PE083	3	first time	PR002	20180121	Large Event	Short Event	Community	NSW	2	95
5	PE100	3	first time	PR002	20180122	Large Event	Short Event	Student	QLD	6	95
6	PE007	4	first time	PR003	20180213	Large Event	Long Event	Student	VIC	6	25
7	PE009	4	first time	PR003	20180130	Large Event	Long Event	Community	WA	2	25
8	PE012	4	first time	PR003	20180213	Large Event	Long Event	Community	NSW	4	5
9	PE068	4	first time	PR003	20180227	Large Event	Long Event	Staff	QLD	9	95
10	PE083	4	first time	PR003	20180213	Large Event	Long Event	Community	NSW	7	70

select count(*) from attendance;



	COUNT(*)
1	5650

select count(*) from AttendanceFact0;



	COUNT(*)
1	5650

The fact table has the same number of rows as the original attendance table. so there is no aggregation happening. we cannot further increase the granularity, so this is the level 0.