MONASH University

MALAYSIA

FIT3003: Business Intelligence and Data Warehousing

MAJOR ASSIGNMENT

Group Number	
MA_05	
Tutorial	
Laboratory-01 (Wednesday 8am)	
Student Name	**Student ID**
Chong Ming Sheng	32202792
Goh Chia Ching	32203004

Table of Contents

GROUP ASSIGNMENT COVER SHEET

Student ID Number	Surname	Given Names
32202792	**Chong**	**Ming Sheng**
32203004	**Goh**	**Chia Ching**

* Please include the names of all other group members.

Unit name and code	FIT3003 Business intelligence and data warehousing	
Title of assignment	Major Assignment	
Lecturer/tutor	Dr Soon Lay Ki	
Tutorial day and time	Weds 8am	**Campus:** Malaysia
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 11 October 2023		**Date submitted** 11 October 2023

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 The material used can be from any source (staff, students or the internet, published and unpublished works).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 .....Chong Ming Sheng..... Date……..9.10.2023.……

Signature .....Goh Chia Ching..... Date……..…9.10.2023……..…

* delete (iii) if not applicable

Signature: *Chong Ming Sheng* Date: *9.10.2023* Signature: *Goh Chia Ching* Date: *9.10.2023*

Privacy Statement

The information on this form is collected for the primary purpose of assessing your assignment and ensuring the academic integrity requirements of the University are met. Other purposes of collection include recording your plagiarism and collusion declaration, attending to course and administrative matters and statistical analyses. If you choose not to complete all the questions on this form it may not be possible for Monash University to assess your assignment. You have a right to access personal information that Monash University holds about you, subject to any exceptions in relevant legislation. If you wish to seek access to your personal information or inquire about the handling of your personal information, please contact the University Privacy Officer: privacyofficer@adm.monash.edu.au

CONTRIBUTION DECLARATION FORM

1. NAME AND CONTRIBUTION DETAILS		
Student ID	**Student Name**	**Contribution %**
32202792	Chong Ming Sheng	50
32203004	Goh Chia Ching	50

2. DECLARATION

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

Signature: Chong Ming Sheng **Date: 9/10/2023**	**Signature: Goh Chia Ching** **Date: 9/10/2023**

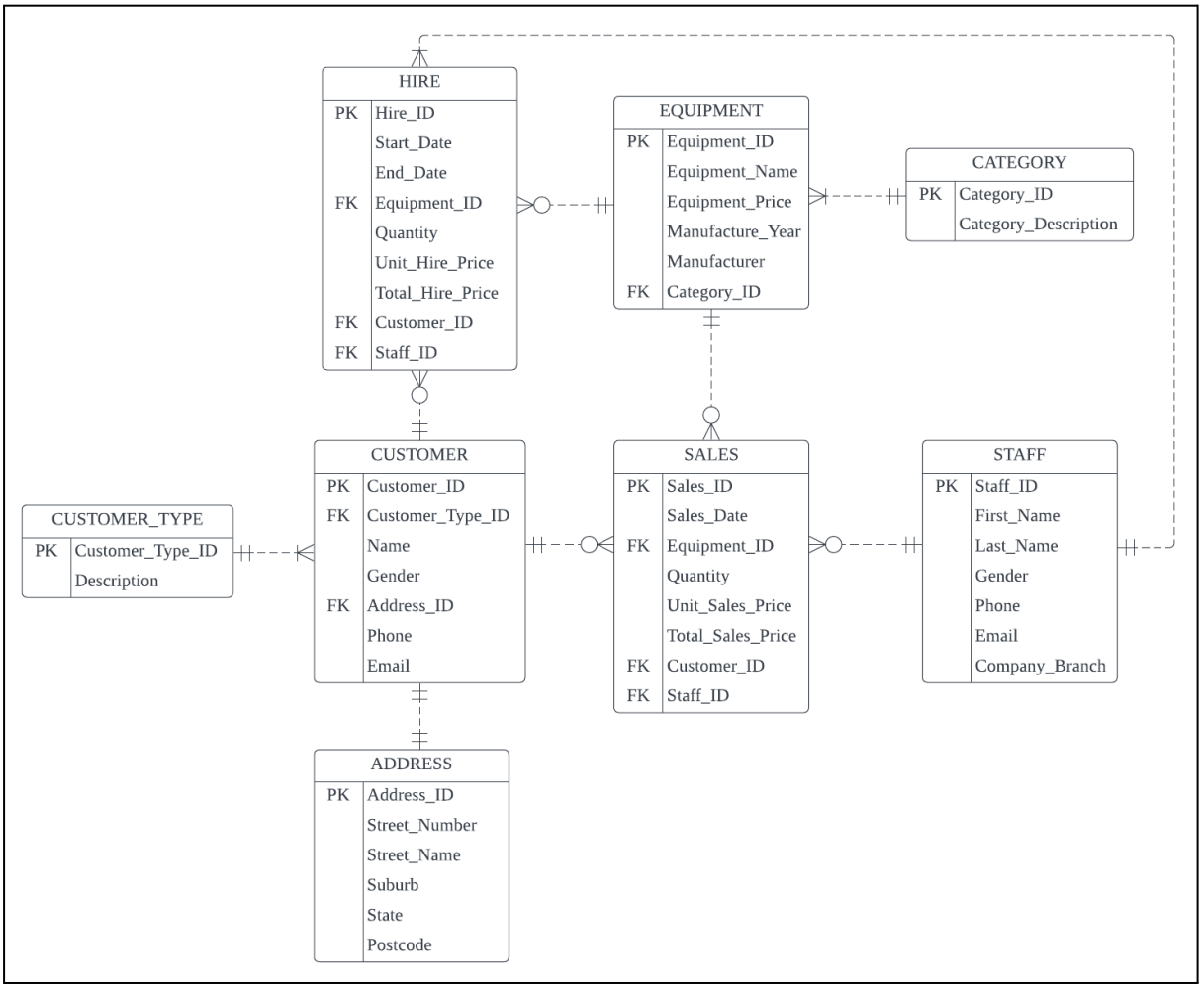
GROUP CONTRIBUTION

No.	Tasks	Completed by
1	E/R Diagram	Ming Sheng, Chia Ching
2	Star Schema 1	Ming Sheng
3	Star Schema 2	Chia Ching
4	Difference between Star Schemas	Chia Ching
5	Explanation of SCD Type	Ming Sheng
6	Table Creation: Star Schema 1	Ming Sheng
7	Table Creation: Star Schema 2	Chia Ching
8	Data Exploration	Ming Sheng
9	Data Cleaning Solutions	Chia Ching

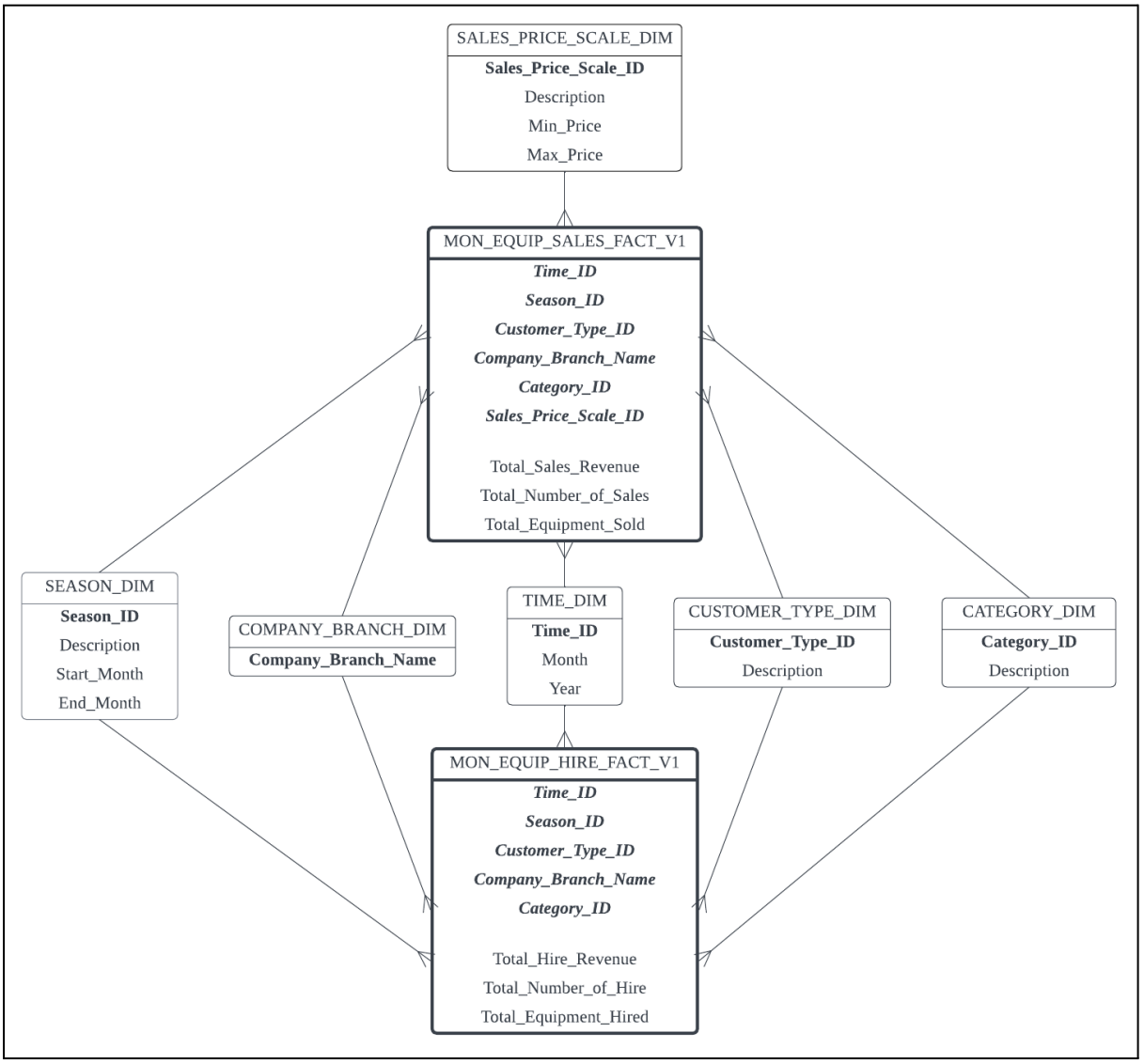
INTRODUCTION

For this assignment, we have designed a data warehouse based on the problem description given as to solve the challenges faced by Monash Equipment Centre (MonEquip). MonEquip's management routinely generates reports to meticulously monitor business facets, including revenue calculations from equipment hiring and sales. In addressing the multifaceted needs of MonEquip, we've undertaken the design of two distinct star schemas, each tailored to different aggregations, thereby enhancing the depth and efficiency of our solution. In addition we have also used various data exploring methods to ensure the integrity of the database, then utilised various methods to clean any inconsistencies. Hence, this report details our methods and results of our efforts which consist of data exploration, data cleaning and creation of two different star schema versions.

E/R DIAGRAM


HIRE
PK Hire_ID
Start_Date
End_Date
FK Equipment_ID
Quantity
Unit_Hire_Price
Total_Hire_Price
FK Customer_ID
FK Staff_ID
EQUIPMENT
PK Equipment_ID
Equipment_Name
Equipment_Price
Manufacture_Year
Manufacturer
FK Category_ID
CATEGORY
PK Category_ID
Category_Description
CUSTOMER
PK Customer_ID
FK Customer_Type_ID
Name
Gender
FK Address_ID
Phone
Email
SALES
PK Sales_ID
Sales_Date
FK Equipment_ID
Quantity
Unit_Sales_Price
Total_Sales_Price
FK Customer_ID
FK Staff_ID
STAFF
PK Staff_ID
First_Name
Last_Name
Gender
Phone
Email
Company_Branch
CUSTOMER_TYPE
PK Customer_Type_ID
Description
ADDRESS
PK Address_ID
Street_Number
Street_Name
Suburb
State
Postcode

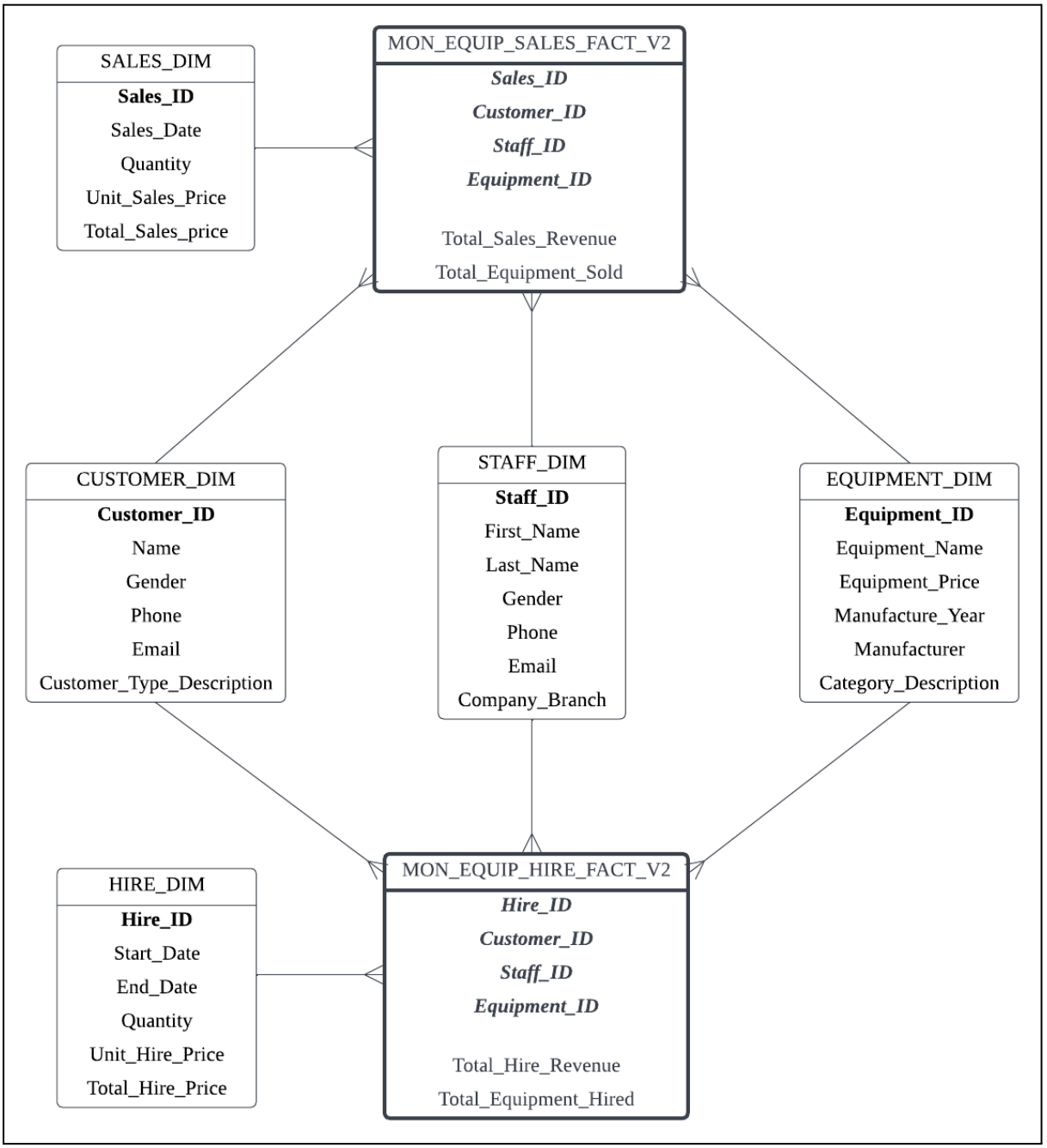
STAR SCHEMA VERSION 1



Assumptions

1. In the **SALES_PRICE_SCALE_DIM**, we assumed that the range for **low sales** is between **0** and **4999** (both inclusive); the range for **medium sales** is between **5000** and **10000** (both inclusive); the range for **high sales** is between **10001** and **999999** (both inclusive), since the maximum TOTAL_SALES_PRICE available in the MonEquip operational database is 338000 (queried via the following SQL statement: *SELECT MAX(TOTAL_SALES_PRICE) FROM MONEQUIP.SALES*).
2. For the Australian Season in **SEASON_DIM**, we assumed that for every year, the **Spring** season will be from **1 September** to **30 November** (both inclusive); **Summer** season will be from **1 December** to **29 February** (both inclusive); **Autumn** season will be from **1 March** to **31 May** (both inclusive); **Winter** season will be from **1 June** to **31 August** (both inclusive).

STAR SCHEMA VERSION 2



DIFFERENCES BETWEEN STAR SCHEMAS

The main difference between the two star schemas is the level of aggregation shown below:

Version 1 (Level 1)	Version 2 (Level 0)
Month is aggregated from the dates in **Hire** and **Sales**	Dates from the **Hire and Sales** tables contain the highest detail
Year is aggregated from the dates in **Hire** and **Sales**	Dates from the **Hire and Sales** tables contain the highest detail
Season is aggregated from the dates in **Hire** and **Sales**	Dates from the **Hire and Sales** tables contain the highest detail
Sales_Price_Scale is aggregated from the revenue from the **Sales** table	The actual prices from the **Sales** table is used instead of Sales_Price_Scale
Company_Branch_Name is aggregated from the **Staff** table	**Staff_DIM** is used to gain access to the company branch
Simplified information on **Equipment** will be referred to from the Category dimension	Information on the **Equipment** is directly accessed from its dimension
There will be 3 fact measures for each fact table because it is pre-aggregated	There will only be 2 fact measures for each fact table as the values of the fact measures removed (i.e. Total_Number_of_Sales and Total_Number_of_Hire) are all 1s

Version 1
This schema is suitable for when we want to analyse and report data at higher levels of aggregation or summarization. Even though the data is less detailed, it is built based on the client's requirements and has better performance. This is because aggregation reduces the amount of data that needs to be processed so this would lead to better query performance.

Version 2
This schema is preferred when you need to maintain detailed information for analysis. This is useful for when we want to look at individual transactions. However, in this case, it would be inconvenient to filter large amounts of data to extract meaningful information, such as **extracting a specific month from the dates in the Hire and Sales tables.** We may need multiple join operations to retrieve aggregated information if needed by the client which may incur a higher cost of operation. Hence, Level-0 star schemas are often not used for data warehousing as they are rather similar to the operational database and contain unnecessary information.

(330 words)

DIFFERENCES BETWEEN SCD TYPES

SCD Type 1

Only the latest value of record is stored by the SCD Type 1, without the historical ones. Therefore, there will be no tracking of historical changes in the value of specific temporal attribute(s). When the attribute value needs to be changed, its old data will be overwritten with the new data. Same object will only correspond to one record.

SCD Type 2

Surrogate Key will be used in the SCD Type 2, in which the original Primary Key is added with a sequence number to distinguish the same object in different time periods. Temporal attributes like Start Date, End Date and Current Flag (indicating current or past record) will also be included in this SCD type. So, unlike SCD Type 1, full history can be kept track by using this SCD type, plus the same object may have multiple records in this case.

SCD Type 3

SCD Type 3 allows us to track changes for only a subset of historical records, for instance Current Equipment Price and Previous Equipment Price that may be applicable in this assignment. Therefore, this SCD type can only track partial historical changes but not full. Similar to SCD Type 1, new data will overwrite old data if there are changes in temporal attribute values, and only one row of record is needed for the same object.

SCD Type 4

Similar to SCD Type 2, the full history of temporal attribute value changes can be kept by using SCD Type 4. However, unlike SCD Type 2, this SCD type will use a new separate table (temporal dimension) to record the data of all temporal attributes, while current data is maintained in the original dimension table. These two tables will be linked together in the star schema. Also, unlike SCD Type 2, Surrogate Key is not required in this SCD type.

SCD Type 6

Combining both SCD Type 2 and Type 3, SCD Type 6 can also record the entire history of changes in temporal attribute values, by using the original dimension table only. Unlike SCD Type 4, no temporal dimension is needed. However, Surrogate Key is not required in this SCD type. Additionally, this SCD type may also have multiple records for the same object. Composite Key is needed to uniquely identify each record in this case.

(338 words)

REASONS FOR THE CHOICE OF SCD TYPE

Star Schema Version 1

We do not use any SCD type here.

In this star schema, the only potential temporal dimension is the COMPANY_BRANCH_DIM, whereas there are no potential attributes spotted. This is because in the real-world scenario, a company branch may have a certain life-time(s) of its business operation, due to the possibility of bankruptcy or renovation.

However, from the MonEquip operational database, the information about the company branch is only available in the STAFF table, reflected by the COMPANY_BRANCH attribute. There is too little information for us to justify if the mentioned dimension is really appropriate to be implemented as a temporal dimension, for example there are no START_DATE and END_DATE attributes signifying such temporal characteristics. So, no SCD type is needed.

Star Schema Version 2

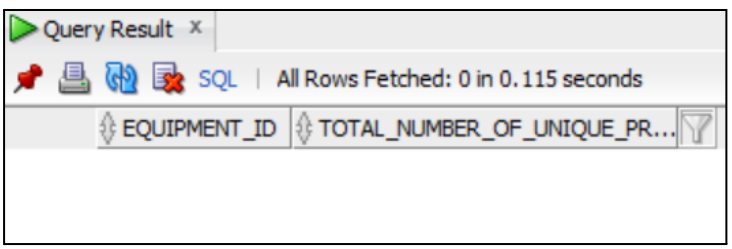
We do not use any SCD type here as well.

In this star schema, there are no potential dimensions spotted, whereas there are potential attributes, namely UNIT_SALES_PRICE, UNIT_HIRE_PRICE and EQUIPMENT_PRICE. This is because in the real-world scenario, these price-related attributes might change over time. For example, different prices could be charged for the same equipment at different times.

However, since both the SALES_ID and HIRE_ID can uniquely identify every sales and hire record, they can perfectly track the full history of the changes in the UNIT_SALES_PRIC and UNIT_HIRE_PRICE if there is any. So, SCD type is not required in this case.

For the EQUIPMENT_PRICE attribute, there is no sign of temporal characteristics found in the MonEquip operational database. To explain, the following SQL statement and query result imply the fact that every equipment has one and only one unique EQUIPMENT_PRICE over the time. It could mean that the price for all equipment is always fixed and never changed. Thus, SCD type is also not required in this case.

```
SELECT
   EQUIPMENT_ID,
   COUNT(DISTINCT EQUIPMENT_PRICE) AS TOTAL_NUMBER_OF_UNIQUE_PRICES
FROM EQUIPMENT_DIM
GROUP BY EQUIPMENT_ID
HAVING COUNT(DISTINCT EQUIPMENT_PRICE) > 1;
```



(308 words)

DATA CLEANING

Based on our Lecture Slides, we have decided to categorise our data issues into 4 types, which are duplication problems, relationship problems, inconsistent and incorrect values, and lastly null value problems. The problems we have found, its evidence and our rectification will be shown below:

Duplication Problems

Between Records

Exploration Code

```
----- ADDRESS TABLE -----

SELECT ADDRESS_ID, COUNT(*)
FROM MONEQUIP.ADDRESS
GROUP BY ADDRESS_ID
HAVING COUNT(*) > 1;

----- CATEGORY TABLE -----

SELECT CATEGORY_ID, COUNT(*)
FROM MONEQUIP.CATEGORY
GROUP BY CATEGORY_ID
HAVING COUNT(*) > 1;

----- CUSTOMER TABLE -----

SELECT CUSTOMER_ID, COUNT(*)
FROM MONEQUIP.CUSTOMER
GROUP BY CUSTOMER_ID
HAVING COUNT(*) > 1;   -----> Problem 1

----- CUSTOMER_TYPE TABLE -----

SELECT CUSTOMER_TYPE_ID, COUNT(*)
FROM MONEQUIP.CUSTOMER_TYPE
GROUP BY CUSTOMER_TYPE_ID
HAVING COUNT(*) > 1;

----- EQUIPMENT TABLE -----

SELECT EQUIPMENT_ID, COUNT(*)
FROM MONEQUIP.EQUIPMENT
GROUP BY EQUIPMENT_ID
HAVING COUNT(*) > 1;

----- HIRE TABLE -----

SELECT HIRE_ID, COUNT(*)
FROM MONEQUIP.HIRE
GROUP BY HIRE_ID
HAVING COUNT(*) > 1;
```

```
----- SALES TABLE -----

SELECT SALES_ID, COUNT(*)
FROM MONEQUIP.SALES
GROUP BY SALES_ID
HAVING COUNT(*) > 1;


----- STAFF TABLE -----

SELECT STAFF_ID, COUNT(*)
FROM MONEQUIP.STAFF
GROUP BY STAFF_ID
HAVING COUNT(*) > 1;
```

Problem 1: Duplicate records for the same CUSTOMER_ID

Before Data Cleaning

```
SELECT CUSTOMER_ID, COUNT(*)
FROM MONEQUIP.CUSTOMER
GROUP BY CUSTOMER_ID
HAVING COUNT(*) > 1;
```

	CUSTOMER_ID	COUNT(*)
1	52	4

```
SELECT CUSTOMER_ID, NAME
FROM MONEQUIP.CUSTOMER
WHERE CUSTOMER_ID=52;
```

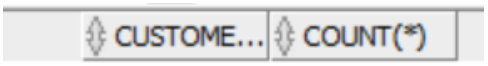
	CUSTOMER_ID	NAME
1	52	Abbie Maddie
2	52	Abbie Maddie
3	52	Abbie Maddie
4	52	Abbie Maddie

Rectification Code

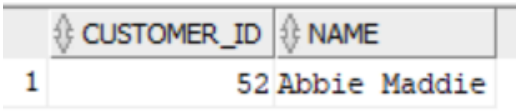
```
CREATE TABLE CUSTOMER AS
   SELECT DISTINCT *
   FROM MONEQUIP.CUSTOMER;
```

After Data Cleaning

```
SELECT CUSTOMER_ID, COUNT(*)
FROM CUSTOMER
GROUP BY CUSTOMER_ID
HAVING COUNT(*) > 1;
```




```
SELECT CUSTOMER_ID, NAME
FROM CUSTOMER
WHERE CUSTOMER_ID=52;
```



Justification

- One primary key can and should only correspond to one unique record
- Having duplicate records is unnecessary and redundant as they contain the same information

Between Attributes

Exploration Code

```
DESCRIBE MONEQUIP.ADDRESS;
DESCRIBE MONEQUIP.CATEGORY;
DESCRIBE MONEQUIP.CUSTOMER;
DESCRIBE MONEQUIP.CUSTOMER_TYPE;
DESCRIBE MONEQUIP.EQUIPMENT;
DESCRIBE MONEQUIP.HIRE;
DESCRIBE MONEQUIP.SALES;
DESCRIBE MONEQUIP.STAFF;
```

No problems found.

Between Tables

Exploration Code

```
SELECT COUNT(*)
FROM ALL_TAB_COLUMNS
WHERE TABLE_NAME = 'CUSTOMER';

DESCRIBE MONEQUIP.CUSTOMER;

SELECT COUNT(*)
FROM ALL_TAB_COLUMNS
WHERE TABLE_NAME = 'CUSTOMER_TYPE';
DESCRIBE MONEQUIP.CUSTOMER_TYPE;
```

No problems found.

Relationship Problems

Invalid FK Values

Exploration Code

```
----- CUSTOMER TABLE -----

SELECT *
FROM MONEQUIP.CUSTOMER
WHERE CUSTOMER_TYPE_ID NOT IN
        (SELECT CUSTOMER_TYPE_ID
        FROM MONEQUIP.CUSTOMER_TYPE);

SELECT *
FROM MONEQUIP.CUSTOMER
WHERE ADDRESS_ID NOT IN
        (SELECT ADDRESS_ID
        FROM MONEQUIP.ADDRESS);


----- EQUIPMENT TABLE -----

SELECT *
FROM MONEQUIP.EQUIPMENT
WHERE CATEGORY_ID NOT IN
        (SELECT CATEGORY_ID
        FROM MONEQUIP.CATEGORY);


----- HIRE TABLE -----

SELECT *
FROM MONEQUIP.HIRE
WHERE EQUIPMENT_ID NOT IN
        (SELECT EQUIPMENT_ID
        FROM MONEQUIP.EQUIPMENT);   -----> Problem 2

SELECT *
FROM MONEQUIP.HIRE
WHERE CUSTOMER_ID NOT IN
        (SELECT CUSTOMER_ID
        FROM MONEQUIP.CUSTOMER);   -----> Problem 3

SELECT *
FROM MONEQUIP.HIRE
WHERE STAFF_ID NOT IN
        (SELECT STAFF_ID
        FROM MONEQUIP.STAFF);   -----> Problem 4


----- SALES TABLE -----

SELECT *
FROM MONEQUIP.SALES
WHERE EQUIPMENT_ID NOT IN
        (SELECT EQUIPMENT_ID
```

```
        FROM MONEQUIP.EQUIPMENT);

SELECT *
FROM MONEQUIP.SALES
WHERE CUSTOMER_ID NOT IN
        (SELECT CUSTOMER_ID
        FROM MONEQUIP.CUSTOMER);

SELECT *
FROM MONEQUIP.SALES
WHERE STAFF_ID NOT IN
        (SELECT STAFF_ID
        FROM MONEQUIP.STAFF);
```

Problem 2: Value of FK is not part of its PKs

Before Data Cleaning

```
SELECT *
FROM MONEQUIP.HIRE
WHERE EQUIPMENT_ID NOT IN
        (SELECT EQUIPMENT_ID
        FROM MONEQUIP.EQUIPMENT);
```

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	301	08/12/2020	08/12/2020	190	1	300	300	181	174

Rectification Code

```
CREATE TABLE HIRE AS
   SELECT *
   FROM MONEQUIP.HIRE;

UPDATE HIRE
SET EQUIPMENT_ID = NULL
WHERE EQUIPMENT_ID NOT IN
   (SELECT EQUIPMENT_ID
    FROM MONEQUIP.EQUIPMENT);
```

After Data Cleaning

```
SELECT *
FROM HIRE
WHERE EQUIPMENT_ID NOT IN
        (SELECT EQUIPMENT_ID
        FROM MONEQUIP.EQUIPMENT);
```

HIRE_ID	START_D...	END_DATE	EQUIPME...	QUANTITY	UNIT_HIR...	TOTAL_HI...	CUSTOME...	STAFF_ID

Justification

- The FK must be referencing to one valid PK
- There is no record for EQUIPMENT_ID of 190 existed in the EQUIPMENT table within the MonEquip database, so this is an invalid record

Problem 3: Value of FK is not part of its PKs

Before Data Cleaning

```
SELECT *
FROM MONEQUIP.HIRE
WHERE CUSTOMER_ID NOT IN
        (SELECT CUSTOMER_ID
        FROM MONEQUIP.CUSTOMER);
```

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	301	08/12/2020	08/12/2020	190	1	300	300	181	174

Rectification Code

```
UPDATE HIRE
SET CUSTOMER_ID = NULL
WHERE CUSTOMER_ID NOT IN
   (SELECT CUSTOMER_ID
    FROM MONEQUIP.CUSTOMER);
```

After Data Cleaning

```
SELECT *
FROM HIRE
WHERE CUSTOMER_ID NOT IN
        (SELECT CUSTOMER_ID
        FROM MONEQUIP.CUSTOMER);
```

HIRE_ID	START_D...	END_DATE	EQUIPME...	QUANTITY	UNIT_HIR...	TOTAL_HI...	CUSTOME...	STAFF_ID

Justification

- The FK must be referencing to one valid PK
- There is no record forCUSTOMER_ID of 181 existed in the CUSTOMER table within the MonEquip operational database, so this is an invalid record

Problem 4: Value of FKs are not part of its PKs

Before Data Cleaning

```
SELECT *
FROM MONEQUIP.HIRE
WHERE STAFF_ID NOT IN
        (SELECT STAFF_ID
        FROM MONEQUIP.STAFF);
```

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	301	08/12/2020	08/12/2020	190	1	300	300	181	174
2	303	25/01/2090	27/12/2099	43	3	50	-150	53	223
3	302	05/12/2020	17/10/2020	21	2	100	200	111	123
4	304	08/12/2020	08/12/2020	114	1	350	-1	34	85

Rectification Code

```
UPDATE HIRE
SET STAFF_ID = NULL
```

```
WHERE STAFF_ID NOT IN
   (SELECT STAFF_ID
    FROM MONEQUIP.STAFF);
```

After Data Cleaning

```
SELECT *
FROM HIRE
WHERE STAFF_ID NOT IN
        (SELECT STAFF_ID
        FROM MONEQUIP.STAFF);
```

HIRE_ID	START_D...	END_DATE	EQUIPME...	QUANTITY	UNIT_HIR...	TOTAL_HI...	CUSTOME...	STAFF_ID

Justification

- The FK must be referencing to one valid PK
- There are no records for STAFF_ID of 174, 223, 123 and 85 existed in the STAFF table within the MONEQUIP operational database, so this is an invalid record

Problem 2 & 3 & 4

Before Data Cleaning

```
SELECT *
FROM HIRE
WHERE EQUIPMENT_ID = NULL OR CUSTOMER_ID = NULL OR STAFF_ID = NULL;
```

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	301	08/12/2020	08/12/2020	(null)	1	300	300	(null)	(null)
2	303	25/01/2090	27/12/2099	43	3	50	543450	53	(null)
3	302	05/12/2020	17/10/2020	21	2	100	200	111	(null)
4	304	08/12/2020	08/12/2020	114	1	350	175	34	(null)

Rectification Code

```
DELETE FROM HIRE
WHERE EQUIPMENT_ID IS NULL OR CUSTOMER_ID IS NULL OR STAFF_ID IS NULL;
```

After Data Cleaning

```
SELECT *
FROM HIRE
WHERE EQUIPMENT_ID = NULL OR CUSTOMER_ID = NULL OR STAFF_ID = NULL;
```

HIRE_ID	START_D...	END_DATE	EQUIPME...	QUANTITY	UNIT_HIR...	TOTAL_HI...	CUSTOME...	STAFF_ID

Justification

- FK value should not be NULL
- Therefore we decided to delete the invalid records

Inconsistent and Incorrect Values

Inconsistent and Incorrect Values
Exploration Code

```
----- ADDRESS TABLE -----

SELECT DISTINCT SUBURB FROM MONEQUIP.ADDRESS ORDER BY SUBURB;
SELECT DISTINCT STATE FROM MONEQUIP.ADDRESS ORDER BY STATE;



----- CATEGORY TABLE -----

SELECT DISTINCT CATEGORY_DESCRIPTION FROM MONEQUIP.CATEGORY
ORDER BY CATEGORY_DESCRIPTION;



----- CUSTOMER TABLE -----

SELECT DISTINCT GENDER FROM MONEQUIP.CUSTOMER;



----- CUSTOMER_TYPE TABLE -----

SELECT DISTINCT DESCRIPTION FROM MONEQUIP.CUSTOMER_TYPE ORDER BY
DESCRIPTION;



----- EQUIPMENT TABLE -----

SELECT DISTINCT EQUIPMENT_NAME FROM MONEQUIP.EQUIPMENT ORDER BY
EQUIPMENT_NAME;
SELECT DISTINCT EQUIPMENT_PRICE FROM MONEQUIP.EQUIPMENT;
SELECT DISTINCT MANUFACTURE_YEAR FROM MONEQUIP.EQUIPMENT;

SELECT DISTINCT MANUFACTURER FROM MONEQUIP.EQUIPMENT ORDER BY
MANUFACTURER;   -----> Problem 5



----- HIRE TABLE -----

SELECT COUNT(*) FROM MONEQUIP.HIRE WHERE QUANTITY < 0;
SELECT COUNT(*) FROM MONEQUIP.HIRE WHERE UNIT_HIRE_PRICE < 0;

SELECT COUNT(*) FROM MONEQUIP.HIRE WHERE TOTAL_HIRE_PRICE < 0;   ----->
Problem 6

SELECT * FROM MONEQUIP.HIRE WHERE TO_CHAR(START_DATE, 'YYYY') > 2023
OR TO_CHAR(END_DATE, 'YYYY') > 2023;   -----> Problem 7

SELECT COUNT(*) FROM MONEQUIP.HIRE WHERE START_DATE > END_DATE;
```

-----> Problem 8

SELECT * FROM MONEQUIP.HIRE WHERE (END_DATE - START_DATE) * QUANTITY * UNIT_HIRE_PRICE <> TOTAL_HIRE_PRICE AND START_DATE < END_DATE;
-----> Problem 9

SELECT * FROM MONEQUIP.HIRE WHERE 0.5 * UNIT_HIRE_PRICE * QUANTITY <> TOTAL_HIRE_PRICE AND START_DATE = END_DATE; -----> Problem 10

----- SALES TABLE -----

SELECT COUNT(*) FROM MONEQUIP.SALES WHERE QUANTITY < 0; -----> Problem 11

SELECT COUNT(*) FROM MONEQUIP.SALES WHERE UNIT_SALES_PRICE < 0;
SELECT COUNT(*) FROM MONEQUIP.SALES WHERE TOTAL_SALES_PRICE < 0;
SELECT DISTINCT QUANTITY FROM MONEQUIP.SALES;
SELECT * FROM MONEQUIP.SALES WHERE TO_CHAR(SALES_DATE, 'YYYY') > 2023;
SELECT COUNT(*) FROM MONEQUIP.SALES WHERE QUANTITY * UNIT_SALES_PRICE <> TOTAL_SALES_PRICE AND QUANTITY > 0;

----- STAFF TABLE -----

SELECT DISTINCT GENDER FROM MONEQUIP.STAFF;
SELECT DISTINCT FIRST_NAME FROM MONEQUIP.STAFF ORDER BY FIRST_NAME;
SELECT DISTINCT LAST_NAME FROM MONEQUIP.STAFF ORDER BY LAST_NAME;
SELECT DISTINCT PHONE FROM MONEQUIP.STAFF ORDER BY PHONE;
SELECT DISTINCT EMAIL FROM MONEQUIP.STAFF ORDER BY EMAIL;
SELECT COUNT(EMAIL) FROM MONEQUIP.STAFF WHERE (EMAIL NOT LIKE '%@%') OR (EMAIL NOT LIKE '%.%');
SELECT DISTINCT COMPANY_BRANCH FROM MONEQUIP.STAFF ORDER BY COMPANY_BRANCH;

Problem 5: Inconsistent letter case for the same string value

Before Data Cleaning

SELECT DISTINCT MANUFACTURER FROM MONEQUIP.EQUIPMENT ORDER BY MANUFACTURER;

	MANUFACTURER
1	Bobcat
2	Case
3	Caterpillar
4	HITACHI
5	Hitachi
6	Isuzu
7	JCB
8	Kenworth
9	Kobelco
10	Komatsu
11	Kubota
12	Volvo
13	Yanmar

Rectification Code

```
CREATE TABLE EQUIPMENT AS
   SELECT *
   FROM MONEQUIP.EQUIPMENT;

UPDATE EQUIPMENT
SET MANUFACTURER = 'Hitachi'
WHERE MANUFACTURER = 'HITACHI';
```

After Data Cleaning

SELECT DISTINCT MANUFACTURER FROM EQUIPMENT ORDER BY MANUFACTURER;

	MANUFACTURER
1	Bobcat
2	Case
3	Caterpillar
4	Hitachi
5	Isuzu
6	JCB
7	Kenworth
8	Kobelco
9	Komatsu
10	Kubota
11	Volvo
12	Yanmar

Justification

- Same manufacturer but having different letter case
- Could be due to inconsistency when inputting data records into the MonEquip operational database

Problem 6: Illogical values for price attribute

Before Data Cleaning

SELECT COUNT(*) FROM MONEQUIP.HIRE WHERE TOTAL_HIRE_PRICE < 0;

	COUNT(*)
1	2

SELECT * FROM MONEQUIP.HIRE WHERE TOTAL_HIRE_PRICE < 0;

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	303	25/01/2090	27/12/2099	43	3	50	-150	53	223
2	304	08/12/2020	08/12/2020	114	1	350	-1	34	85

Rectification Code

```
UPDATE HIRE
SET TOTAL_HIRE_PRICE = (END_DATE - START_DATE) * UNIT_HIRE_PRICE *
QUANTITY
WHERE TOTAL_HIRE_PRICE < 0 AND END_DATE > START_DATE;

UPDATE HIRE
SET TOTAL_HIRE_PRICE = 0.5 * UNIT_HIRE_PRICE * QUANTITY
WHERE TOTAL_HIRE_PRICE < 0 AND END_DATE = START_DATE;
```

After Data Cleaning

SELECT * FROM HIRE WHERE TOTAL_HIRE_PRICE < 0;

HIRE_ID	START_D...	END_DATE	EQUIPME...	QUANTITY	UNIT_HIR...	TOTAL_HI...	CUSTOME...	STAFF_ID

Justification

- Illogical to have negative values for price
- Could be due to human error when inputting records
- Therefore we decided to correct the values based on the formula given

Problem 7: Illogical starting and ending dates in the future time

Before Data Cleaning

SELECT * FROM MONEQUIP.HIRE WHERE TO_CHAR(START_DATE, 'YYYY') > 2023 OR TO_CHAR(END_DATE, 'YYYY') > 2023;

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	303	25/01/2090	27/12/2099	43	3	50	-150	53	223

Rectification Code

```
DELETE FROM HIRE
WHERE TO_CHAR(START_DATE, 'YYYY') > 2023 OR TO_CHAR(END_DATE, 'YYYY') >
2023;
```

After Data Cleaning

SELECT * FROM HIRE WHERE TO_CHAR(START_DATE, 'YYYY') > 2023 OR TO_CHAR(END_DATE, 'YYYY') > 2023;

HIRE_ID	START_D...	END_DATE	EQUIPME...	QUANTITY	UNIT_HIR...	TOTAL_HI...	CUSTOME...	STAFF_ID

Justification

- Illogical to have records with the starting and ending dates in the future time
- Could be due to human error when inputting records
- Therefore we decided to delete the invalid records

Problem 8: Ending date is earlier than starting date

Before Data Cleaning

SELECT COUNT(*) FROM MONEQUIP.HIRE WHERE START_DATE > END_DATE;

	COUNT(*)
1	1

SELECT * FROM MONEQUIP.HIRE WHERE START_DATE > END_DATE;

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	302	05/12/2020	17/10/2020	21	2	100	200	111	123

Rectification Code

DELETE FROM HIRE
WHERE START_DATE > END_DATE;

After Data Cleaning

SELECT * FROM HIRE WHERE START_DATE > END_DATE;

HIRE_ID	START_D...	END_DATE	EQUIPME...	QUANTITY	UNIT_HIR...	TOTAL_HI...	CUSTOME...	STAFF_ID

Justification

- Illogica to have ending date earlier than starting date
- Could be due to human error when inputting records
- So we decided to delete the invalid records

Problem 9: Incorrect calculations based on formula given

Before Data Cleaning

SELECT * FROM MONEQUIP.HIRE WHERE (END_DATE - START_DATE) * QUANTITY * UNIT_HIRE_PRICE <> TOTAL_HIRE_PRICE AND START_DATE < END_DATE;

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	1	11/05/2018	14/05/2018	135	3	80	240	77	2
2	2	17/05/2018	20/05/2018	49	2	660	1320	70	38
3	4	21/05/2018	25/05/2018	36	1	540	540	124	41
4	5	21/05/2018	23/05/2018	37	2	600	1200	87	35
5	6	22/05/2018	26/05/2018	53	2	500	1000	15	31
6	7	24/05/2018	28/05/2018	73	3	140	420	77	10
7	8	25/05/2018	28/05/2018	135	3	80	240	140	40
8	11	12/06/2018	14/06/2018	71	2	450	900	145	21
9	18	12/07/2018	14/07/2018	25	3	600	1800	114	3
10	24	27/07/2018	29/07/2018	99	2	20	40	63	5
11	25	27/07/2018	29/07/2018	147	1	220	220	19	2
12	26	30/07/2018	03/08/2018	22	3	450	1350	113	27
13	27	01/08/2018	03/08/2018	10	2	360	720	99	47
14	28	01/08/2018	05/08/2018	133	2	600	1200	126	28
15	32	09/08/2018	13/08/2018	155	2	400	800	4	4

Rectification Code

```
UPDATE HIRE
SET TOTAL_HIRE_PRICE = (END_DATE - START_DATE) * UNIT_HIRE_PRICE *
QUANTITY
WHERE (END_DATE - START_DATE) * QUANTITY * UNIT_HIRE_PRICE <>
TOTAL_HIRE_PRICE AND START_DATE < END_DATE;
```

After Data Cleaning

```
SELECT * FROM MONEQUIP.HIRE WHERE (END_DATE - START_DATE) * QUANTITY
* UNIT_HIRE_PRICE <> TOTAL_HIRE_PRICE AND START_DATE < END_DATE;
```

HIRE_ID	START_D...	END_DATE	EQUIPME...	QUANTITY	UNIT_HIR...	TOTAL_HI...	CUSTOME...	STAFF_ID

```
SELECT * FROM HIRE WHERE START_DATE < END_DATE;
```

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	1	11/05/2018	14/05/2018	135	3	80	720	77	2
2	2	17/05/2018	20/05/2018	49	2	660	3960	70	38
3	3	18/05/2018	19/05/2018	117	1	150	150	58	17
4	4	21/05/2018	25/05/2018	36	1	540	2160	124	41
5	5	21/05/2018	23/05/2018	37	2	600	2400	87	35
6	6	22/05/2018	26/05/2018	53	2	500	4000	15	31
7	7	24/05/2018	28/05/2018	73	3	140	1680	77	10
8	8	25/05/2018	28/05/2018	135	3	80	720	140	40
9	11	12/06/2018	14/06/2018	71	2	450	1800	145	21
10	13	24/06/2018	25/06/2018	61	3	150	450	114	10
11	15	30/06/2018	01/07/2018	9	1	360	360	47	8
12	17	07/07/2018	08/07/2018	85	2	250	500	134	44
13	18	12/07/2018	14/07/2018	25	3	600	3600	114	3
14	19	13/07/2018	14/07/2018	112	1	300	300	135	47
15	20	13/07/2018	14/07/2018	141	3	80	240	71	28

```
-- Show the first 5 records where problem was found before data cleaning
-- Verify if the TOTAL_HIRE_PRICE has been updated successfully
SELECT * FROM HIRE WHERE START_DATE < END_DATE AND (HIRE_ID = 1 OR
HIRE_ID = 2 OR HIRE_ID = 4 OR HIRE_ID = 5 OR HIRE_ID = 6);
```

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	1	11/05/2018	14/05/2018	135	3	80	720	77	2
2	2	17/05/2018	20/05/2018	49	2	660	3960	70	38
3	4	21/05/2018	25/05/2018	36	1	540	2160	124	41
4	5	21/05/2018	23/05/2018	37	2	600	2400	87	35
5	6	22/05/2018	26/05/2018	53	2	500	4000	15	31

Justification

- Incorrect TOTAL_HIRE_PRICE calculation based on the given formula (if the equipment is not returned within the same day)
- So we decided to correct the TOTAL_HIRE_PRICE values

Problem 10: Incorrect calculations based on formula given

Before Data Cleaning

SELECT * FROM MONEQUIP.HIRE WHERE 0.5 * UNIT_HIRE_PRICE * QUANTITY <> TOTAL_HIRE_PRICE AND START_DATE = END_DATE;

	HI...	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	301	08/12/2020	08/12/2020	190	1	300	300	181	174
2	304	08/12/2020	08/12/2020	114	1	350	-1	34	85

Rectification Code

UPDATE HIRE
SET TOTAL_HIRE_PRICE = 0.5 * UNIT_HIRE_PRICE * QUANTITY
WHERE 0.5 * UNIT_HIRE_PRICE * QUANTITY <> TOTAL_HIRE_PRICE AND START_DATE = END_DATE;

After Data Cleaning

SELECT * FROM HIRE WHERE 0.5 * UNIT_HIRE_PRICE * QUANTITY <> TOTAL_HIRE_PRICE AND START_DATE = END_DATE;

HIRE_ID	START_D...	END_DATE	EQUIPME...	QUANTITY	UNIT_HIR...	TOTAL_HI...	CUSTOME...	STAFF_ID

SELECT * FROM HIRE WHERE START_DATE = END_DATE;

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	9	28/05/2018	28/05/2018	127	2	170	170	8	46
2	10	29/05/2018	29/05/2018	86	2	200	200	5	28
3	12	15/06/2018	15/06/2018	23	2	360	360	74	43
4	14	25/06/2018	25/06/2018	85	3	210	315	110	44
5	16	05/07/2018	05/07/2018	74	1	280	140	129	7
6	30	04/08/2018	04/08/2018	64	1	220	110	79	34
7	36	14/08/2018	14/08/2018	151	2	100	100	10	38
8	39	19/08/2018	19/08/2018	67	2	300	300	124	19
9	40	21/08/2018	21/08/2018	16	3	80	120	126	33
10	43	25/08/2018	25/08/2018	2	3	210	315	140	27
11	45	01/09/2018	01/09/2018	95	1	160	80	101	14
12	52	19/09/2018	19/09/2018	15	3	70	105	111	1
13	59	20/10/2018	20/10/2018	45	2	160	160	123	27
14	64	29/10/2018	29/10/2018	138	1	90	45	13	36
15	77	04/12/2018	04/12/2018	101	1	80	40	24	10

-- Show the records where problem was found before data cleaning
-- Verify if the TOTAL_HIRE_PRICE has been updated successfully
SELECT * FROM HIRE WHERE START_DATE = END_DATE AND (HIRE_ID = 301 OR HIRE_ID = 304);

	HIRE_ID	START_DATE	END_DATE	EQUIPMENT_ID	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE	CUSTOMER_ID	STAFF_ID
1	301	08/12/2020	08/12/2020	190	1	300	150	181	174
2	304	08/12/2020	08/12/2020	114	1	350	175	34	85

Justification

- Incorrect TOTAL_HIRE_PRICE calculation based on the given formula (if the equipment is returned within the same day)
- So we decided to correct the TOTAL_HIRE_PRICE values

Problem 11: Illogical value for quantity attribute

Before Data Cleaning

SELECT COUNT(*) FROM MONEQUIP.SALES WHERE QUANTITY < 0;

	COUNT(*)
1	1

SELECT * FROM MONEQUIP.SALES WHERE QUANTITY < 0;

	SALES_ID	SALES_DATE	EQUIPMENT_ID	QUANTITY	UNIT_SALES_PRICE	TOTAL_SALES_PRICE	CUSTOMER_ID	STAFF_ID
1	151	15/12/2020	20	-3	45500	182000	2	37

Rectification Code

```
CREATE TABLE SALES AS
   SELECT * FROM MONEQUIP.SALES;

UPDATE SALES
SET QUANTITY = TOTAL_SALES_PRICE / UNIT_SALES_PRICE
WHERE QUANTITY < 0;
```

After Data Cleaning

SELECT * FROM SALES WHERE QUANTITY < 0;

SALES_ID	SALES_D...	EQUIPME...	QUANTITY	UNIT_SAL...	TOTAL_S...	CUSTOME...	STAFF_ID

Justification

- Illogical to have negative quantity
- Could be due to human error when inputting records
- So we decided to correct the QUANTITY value by using the formula: QUANTITY = TOTAL_SALES_PRICE / UNIT_SALES_PRICE

Null Value Problems

Exploration Code

```
----- ADDRESS TABLE -----

DESCRIBE MONEQUIP.ADDRESS;
SELECT COUNT(*) FROM MONEQUIP.ADDRESS WHERE STREET_NAME = '';
SELECT COUNT(*) FROM MONEQUIP.ADDRESS WHERE STREET_NAME = ' ';
SELECT COUNT(*) FROM MONEQUIP.ADDRESS WHERE UPPER(STREET_NAME) = 'NULL';
SELECT COUNT(*) FROM MONEQUIP.ADDRESS WHERE SUBURB = '';
SELECT COUNT(*) FROM MONEQUIP.ADDRESS WHERE SUBURB = ' ';
SELECT COUNT(*) FROM MONEQUIP.ADDRESS WHERE UPPER(SUBURB) = 'NULL';
SELECT COUNT(*) FROM MONEQUIP.ADDRESS WHERE STATE = '';
SELECT COUNT(*) FROM MONEQUIP.ADDRESS WHERE STATE = ' ';
SELECT COUNT(*) FROM MONEQUIP.ADDRESS WHERE UPPER(STATE) = 'NULL';


----- CATEGORY TABLE -----

DESCRIBE MONEQUIP.CATEGORY;
SELECT COUNT(*) FROM MONEQUIP.CATEGORY WHERE CATEGORY_DESCRIPTION = '';
SELECT COUNT(*) FROM MONEQUIP.CATEGORY WHERE CATEGORY_DESCRIPTION = ' ';

SELECT COUNT(*) FROM MONEQUIP.CATEGORY WHERE
UPPER(CATEGORY_DESCRIPTION) = 'NULL';   -----> Problem 12


----- CUSTOMER TABLE -----

DESCRIBE MONEQUIP.CUSTOMER;
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE NAME = '';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE NAME = ' ';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE UPPER(NAME) = 'NULL';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE GENDER = '';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE GENDER = ' ';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE UPPER(GENDER) = 'NULL';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE PHONE = '';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE PHONE = ' ';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE UPPER(PHONE) = 'NULL';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE EMAIL = '';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE EMAIL = ' ';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER WHERE UPPER(EMAIL) = 'NULL';


----- CUSTOMER_TYPE TABLE -----

DESCRIBE MONEQUIP.CUSTOMER_TYPE;
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER_TYPE WHERE DESCRIPTION = '';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER_TYPE WHERE DESCRIPTION = ' ';
SELECT COUNT(*) FROM MONEQUIP.CUSTOMER_TYPE WHERE UPPER(DESCRIPTION) =
'NULL';
```

```
----- EQUIPMENT TABLE -----

DESCRIBE MONEQUIP.EQUIPMENT;
SELECT COUNT(*) FROM MONEQUIP.EQUIPMENT WHERE EQUIPMENT_NAME = '';
SELECT COUNT(*) FROM MONEQUIP.EQUIPMENT WHERE EQUIPMENT_NAME = ' ';
SELECT COUNT(*) FROM MONEQUIP.EQUIPMENT WHERE UPPER(EQUIPMENT_NAME) =
'NULL';
SELECT COUNT(*) FROM MONEQUIP.EQUIPMENT WHERE MANUFACTURER = '';
SELECT COUNT(*) FROM MONEQUIP.EQUIPMENT WHERE MANUFACTURER = ' ';
SELECT COUNT(*) FROM MONEQUIP.EQUIPMENT WHERE UPPER(MANUFACTURER) =
'NULL';


----- HIRE TABLE -----

DESCRIBE MONEQUIP.HIRE;
DESCRIBE MONEQUIP.SALES;


----- STAFF TABLE -----

DESCRIBE MONEQUIP.STAFF;
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE FIRST_NAME = '';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE FIRST_NAME = ' ';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE UPPER(FIRST_NAME) = 'NULL';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE LAST_NAME = '';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE LAST_NAME = ' ';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE UPPER(LAST_NAME) = 'NULL';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE GENDER = '';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE GENDER = ' ';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE UPPER(GENDER) = 'NULL';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE PHONE = '';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE PHONE = ' ';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE UPPER(PHONE) = 'NULL';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE EMAIL = '';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE EMAIL = ' ';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE UPPER(EMAIL) = 'NULL';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE COMPANY_BRANCH = '';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE COMPANY_BRANCH = ' ';
SELECT COUNT(*) FROM MONEQUIP.STAFF WHERE UPPER(COMPANY_BRANCH) = 'NULL';
```

Problem 12: Meaningless ‘NULL’ value for description attribute

Before Data Cleaning

```
SELECT COUNT(*) FROM MONEQUIP.CATEGORY WHERE
UPPER(CATEGORY_DESCRIPTION) = 'NULL';
```

	COUNT(*)
1	1

SELECT * FROM MONEQUIP.CATEGORY WHERE
UPPER(CATEGORY_DESCRIPTION) = 'NULL';

	CATEGORY_ID	CATEGORY_DESCRIPTION
1	15	null

-- Find for corresponding record(s) in MONEQUIP.EQUIPMENT
SELECT * FROM MONEQUIP.EQUIPMENT WHERE CATEGORY_ID = (SELECT
CATEGORY_ID FROM MONEQUIP.CATEGORY WHERE
UPPER(CATEGORY_DESCRIPTION) = 'NULL');

	EQUIPMENT_ID	EQUIPMENT_NAME	EQUIPMENT_PRICE	MANUFACTURE_YEAR	MANUFACTURER	CATEGORY_ID
1	158	EXCAVATOR - POST HOLE ATTACHMENT SUIT 3.5T	12200	2017	HITACHI	15

Rectification Code

```
CREATE TABLE CATEGORY AS
   SELECT DISTINCT *
   FROM MONEQUIP.CATEGORY;

DELETE FROM CATEGORY
WHERE UPPER(CATEGORY_DESCRIPTION) = 'NULL';

-- Delete the corresponding record(s) from EQUIPMENT
DELETE FROM EQUIPMENT
WHERE CATEGORY_ID =
   (SELECT CATEGORY_ID
    FROM MONEQUIP.CATEGORY
    WHERE UPPER(CATEGORY_DESCRIPTION) = 'NULL');
```

After Data Cleaning

SELECT * FROM CATEGORY WHERE UPPER(CATEGORY_DESCRIPTION) = 'NULL';

CATEGOR...	CATEGOR...

Justification

- It is meaningless to have CATEGORY_DESCRIPTION value being 'NULL'
- So we decided to delete the corresponding invalid records from both CATEGORY and EQUIPMENT tables

STRATEGIES USED IN DATA CLEANING PROCESS

To ensure the quality and the reliability of the database before we migrate it to the data warehouse, we have employed a comprehensive set of strategies for data exploration and cleaning. The goal was to identify and rectify various data issues, ranging from duplication problems to relationship issues, null value problems, and data inconsistencies. Our approach to data exploration and cleaning was systematic, addressing each problem type one by one, starting with duplication problems, followed by relationship issues, null value problems, and data inconsistencies. We tackled each type of issue systematically and conducted exploration and cleaning on a table-by-table basis to ensure adherence to required standards and relationships.

Duplication Problems

Between Records

To address duplication between records, we employed SQL queries and tools to identify duplicate entries based on unique identifiers (primary keys). The resolution involved deciding whether to retain the first or last occurrence or merge information based on the context.

Between Attributes

For duplication between attributes within a single column, we utilised SQL aggregation functions to identify and subsequently address duplicate values. The resolution depended on the context, with an emphasis on consolidating values where necessary.

Between Tables

To tackle duplication issues between tables, we utilised comparison of unique keys across tables. This process allowed us to ensure that foreign key relationships were correctly defined and to identify and rectify any mismatches.

Relationship Problems

Invalid FK Values

Identification of invalid foreign key values was conducted through SQL queries. The resolution involved correcting or removing invalid foreign key values to maintain referential integrity.

Inconsistent and Incorrect Values

We addressed inconsistencies in values at the record level and between attributes through careful inspection using SQL queries. This process allowed us to identify and rectify inconsistencies, ensuring data accuracy and logical consistency.

Null Value Problems

Attribute Level

For null values at the attribute level, we identified records with missing values through SQL queries. The resolution involved imputing values, removing records, or filling nulls with appropriate defaults.

Between Records

Null values between records were identified using SQL queries. The resolution included deciding whether to impute values, remove records, or fill nulls with defaults, ensuring completeness and accuracy.

Between Attributes

To address null value problems between attributes, we examined columns for inconsistent use of nulls. The resolution involved establishing consistent rules for nulls and considering filling with defaults or valid values.

Data Inconsistency

Focus on Specific Data Types

Date (DATE)

For date-related inconsistencies, we utilised SQL queries to identify future dates or inconsistent date formats. The resolution involved standardising date formats and examining future dates for accuracy.

String (CHAR & VARCHAR2)

In the case of string-related issues, we employed SQL queries to identify anomalies such as upper('null'), empty strings, or unnecessary spaces. The resolution included standardising string formats and removing or replacing misleading values.

Number (NUMBER)

To address numeric data inconsistencies, we used SQL queries to identify illogical negative values or inconsistent use of numeric types. The resolution involved correcting negative values or ensuring the appropriate use of numeric data types.

Conclusion

Through the strategies we employed, we have significantly enhanced the quality and reliability of our data in the data warehouse. Regular monitoring and maintenance will be essential to sustain this level of data quality over time. This is to ensure that the information stored is not only accurate but also a reliable foundation for decision-making processes.

SCREENSHOTS OF IMPLEMENTATION & TABLES CREATED

Star Schema Version 1

```
-- SALES_PRICE_SCALE Dimension

CREATE TABLE SALES_PRICE_SCALE_DIM
    (
        Sales_Price_Scale_ID NUMBER(5),
        Description VARCHAR2(50),
        Min_Price NUMBER(10),
        Max_Price NUMBER(10)
    );

-- Insert records into SALES_PRICE_SCALE Dimension

INSERT INTO SALES_PRICE_SCALE_DIM VALUES (1, 'Low sales', 0, 4999);
INSERT INTO SALES_PRICE_SCALE_DIM VALUES (2, 'Medium sales', 5000, 10000);
INSERT INTO SALES_PRICE_SCALE_DIM VALUES (3, 'High sales', 10001, 999999);
```

	SALES_PRICE_SCALE_ID	DESCRIPTION	MIN_PRICE	MAX_PRICE
1	1	Low sales	0	4999
2	2	Medium sales	5000	10000
3	3	High sales	10001	999999

```
-- SEASON Dimension

CREATE TABLE SEASON_DIM
    (
        Season_ID NUMBER(1),
        Description VARCHAR2(30),
        Start_Month VARCHAR2(2),
        End_Month VARCHAR2(2)
    );

-- Insert records into SEASON Dimension

INSERT INTO SEASON_DIM VALUES (1, 'Spring', '09', '11');
INSERT INTO SEASON_DIM VALUES (2, 'Summer', '12', '02');
INSERT INTO SEASON_DIM VALUES (3, 'Autumn', '03', '05');
INSERT INTO SEASON_DIM VALUES (4, 'Winter', '06', '08');
```

	SEASON_ID	DESCRIPTION	START_MONTH	END_MONTH
1	1	Spring	09	11
2	2	Summer	12	02
3	3	Autumn	03	05
4	4	Winter	06	08

```
-- COMPANY_BRANCH Dimension

CREATE TABLE COMPANY_BRANCH_DIM AS
    SELECT DISTINCT Company_Branch AS Company_Branch_Name
    FROM MONEQUIP.STAFF
    ORDER BY Company_Branch_Name;
```

	COMPANY_BRANCH_NAME
1	Caulfield
2	Chadstone
3	Cheltenham
4	Clayton
5	Dandenong
6	Docklands
7	Eltham
8	Fitzroy
9	Geelong
10	Hughesdale
11	Pakenham
12	Parkville
13	Prahran
14	Richmond
15	Toorak

```
-- Temporary TIME Dimension

CREATE TABLE TIME_DIM_TEMP
(
    TIme_ID VARCHAR2(6)
);

-- Insert records into temporary TIME Dimension

INSERT INTO TIME_DIM_TEMP
SELECT DISTINCT TO_CHAR(Sales_Date, 'YYYYMM') AS Time_ID
FROM SALES;

INSERT INTO TIME_DIM_TEMP
SELECT DISTINCT TO_CHAR(Start_Date, 'YYYYMM') AS Time_ID
FROM HIRE;
```

	TIME_ID
1	201905
2	201908
3	202005
4	201812
5	201901
6	201903
7	201909
8	202007
9	202008
10	202010
11	201805
12	201809
13	201811
14	201902
15	201904
16	201907
17	201912
18	202009
19	201807
20	201808

```
-- TIME Dimension

CREATE TABLE TIME_DIM
(
    Time_ID VARCHAR2(6),
    Month VARCHAR2(2),
    Year VARCHAR2(4)
);

-- Insert records into TIME Dimension

INSERT INTO TIME_DIM
SELECT
    DISTINCT Time_ID,
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM'),
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'YYYY')
FROM TIME_DIM_TEMP;
```

	TIME_ID	MONTH	YEAR
1	202010	10	2020
2	202009	09	2020
3	201906	06	2019
4	201810	10	2018
5	201901	01	2019
6	202004	04	2020
7	201806	06	2018
8	202008	08	2020
9	201811	11	2018
10	201902	02	2019
11	202001	01	2020
12	202011	11	2020
13	201905	05	2019
14	201908	08	2019
15	201909	09	2019
16	202007	07	2020
17	202002	02	2020
18	201910	10	2019
19	201809	09	2018
20	201807	07	2018

```
-- CUSTOMER_TYPE Dimension

CREATE TABLE CUSTOMER_TYPE_DIM AS
    SELECT *
    FROM MONEQUIP.CUSTOMER_TYPE
    ORDER BY CUSTOMER_TYPE_ID;
```

	CUSTOMER_TYPE_ID	DESCRIPTION
1	1	Individual
2	2	Business

```
-- CATEGORY Dimension

CREATE TABLE CATEGORY_DIM AS
    SELECT
        CATEGORY_ID,
        CATEGORY_DESCRIPTION AS Description
    FROM CATEGORY
    ORDER BY CATEGORY_ID;
```

	CATEGORY_ID	DESCRIPTION
1	1	Access
2	2	Air Compressor
3	3	Compaction
4	4	Concrete
5	5	Earthmoving
6	6	Generators
7	7	Landscaping
8	8	Lighting
9	9	Plumbing
10	10	Rail
11	11	Safety
12	12	Site Equipment
13	13	Trailers
14	14	Vehicles

```
-- MON_EQUIP_SALES Temporary Fact Table V1

CREATE TABLE MON_EQUIP_SALES_FACT_TEMP_V1 AS
    SELECT
        TO_CHAR(S.Sales_Date, 'YYYYMM') AS Time_ID,
        C.Customer_Type_ID,
        F.Company_Branch AS Company_Branch_Name,
        Y.Category_ID,
        S.Total_Sales_Price,
        S.Sales_ID,
        S.Quantity
    FROM
        SALES S,
        CUSTOMER C,
        MONEQUIP.STAFF F,
        EQUIPMENT E,
        CATEGORY Y
    WHERE
        S.Customer_ID = C.Customer_ID AND
        S.Staff_ID = F.Staff_ID AND
        S.Equipment_ID = E.Equipment_ID AND
        E.Category_ID = Y.Category_ID;

-- Add new attributes

ALTER TABLE MON_EQUIP_SALES_FACT_TEMP_V1
ADD (Season_ID NUMBER(1));

ALTER TABLE MON_EQUIP_SALES_FACT_TEMP_V1
ADD (Sales_Price_Scale_ID NUMBER(10));

-- Update value of new attribute (Season_ID)

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Season_ID = 1
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '09' AND
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '11';

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Season_ID = 2
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '12' OR
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '02';

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Season_ID = 3
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '03' AND
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '05';

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Season_ID = 4
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '06' AND
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '08';
```

```
-- Update value of new attribute (Sales_Price_Scale_ID)

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Sales_Price_Scale_ID = 1
WHERE Total_Sales_Price BETWEEN 0 AND 4999;

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Sales_Price_Scale_ID = 2
WHERE Total_Sales_Price BETWEEN 5000 AND 10000;

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Sales_Price_Scale_ID = 3
WHERE Total_Sales_Price BETWEEN 10001 AND 999999;
```

	TIME_ID	CUSTOMER_TYPE_ID	COMPANY_BRANCH_NAME	CATEGORY_ID	TOTAL_SALES_PRICE	SALES_ID	QUANTITY	SEASON_ID	SALES_PRICE_SCALE_ID
1	201810	1	Toorak	1	63000	34	2	1	3
2	202009	1	Geelong	1	94500	131	3	1	3
3	201910	2	Clayton	1	154050	79	3	1	3
4	201904	1	Toorak	1	156000	57	4	3	3
5	201809	2	Docklands	1	214500	30	3	1	3
6	201909	1	Clayton	1	88800	76	2	1	3
7	201905	1	Richmond	2	29400	61	3	3	3
8	202005	1	Caulfield	2	130000	113	4	3	3
9	201911	2	Pakenham	2	21600	85	1	1	3
10	201911	2	Clayton	2	91000	84	2	1	3
11	202006	2	Chadstone	2	91000	121	2	4	3
12	202012	2	Pakenham	2	182000	150	4	2	3
13	202012	2	Pakenham	2	182000	151	4	2	3
14	201908	1	Pakenham	2	90000	71	2	4	3
15	201903	2	Clayton	2	150000	53	2	3	3

```
-- MON_EQUIP_SALES Fact Table V1

CREATE TABLE MON_EQUIP_SALES_FACT_V1 AS
    SELECT
        Time_ID,
        Season_ID,
        Customer_Type_ID,
        Company_Branch_Name,
        Category_ID,
        Sales_Price_Scale_ID,
        SUM(Total_Sales_Price) AS Total_Sales_Revenue,
        COUNT(Sales_ID) AS Total_Number_of_Sales,
        SUM(Quantity) AS Total_Equipment_Sold
    FROM
        MON_EQUIP_SALES_FACT_TEMP_V1
    GROUP BY
        Time_ID,
        Season_ID,
        Customer_Type_ID,
        Company_Branch_Name,
        Category_ID,
        Sales_Price_Scale_ID;
```

	TIME_ID	SEASON_ID	CUSTOMER_TYPE_ID	COMPANY_BRANCH_NAME	CATEGORY_ID	SALES_PRICE_SCALE_ID	TOTAL_SALES_REVENUE	TOTAL_NUMBER_OF_SALES	TOTAL_EQUIPMENT_SOLD
1	201808	4	1	Parkville	3	3	65000	1	1
2	202009	1	1	Pakenham	4	3	40000	1	4
3	201807	4	1	Prahran	5	3	162000	1	3
4	202006	4	1	Clayton	5	3	240000	1	4
5	201909	1	2	Parkville	6	3	120000	1	4
6	201809	1	2	Clayton	7	3	19200	1	2
7	201808	4	2	Caulfield	9	2	5600	1	2
8	202006	4	2	Caulfield	9	3	138600	1	3
9	201809	1	2	Eltham	9	3	11200	1	2
10	201903	3	2	Hughesdale	12	2	5000	1	1
11	202010	1	1	Clayton	14	3	72000	1	3
12	201911	1	2	Clayton	2	3	91000	1	2
13	202006	4	2	Chadstone	2	3	91000	1	2
14	201908	4	2	Clayton	3	3	18000	1	1
15	201807	4	2	Clayton	5	3	338000	1	4

```
-- MON_EQUIP_HIRE Temporary Fact Table V1

CREATE TABLE MON_EQUIP_HIRE_FACT_TEMP_V1 AS
    SELECT
        TO_CHAR(H.Start_Date, 'YYYYMM') AS Time_ID,
        C.Customer_Type_ID,
        F.Company_Branch AS Company_Branch_Name,
        Y.Category_ID,
        H.Total_Hire_Price,
        H.Hire_ID,
        H.Quantity
    FROM
        HIRE H,
        CUSTOMER C,
        MONEQUIP.STAFF F,
        EQUIPMENT E,
        CATEGORY Y
    WHERE
        H.Customer_ID = C.Customer_ID AND
        H.Staff_ID = F.Staff_ID AND
        H.Equipment_ID = E.Equipment_ID AND
        E.Category_ID = Y.Category_ID;

-- Add new attributes

ALTER TABLE MON_EQUIP_HIRE_FACT_TEMP_V1
ADD (Season_ID NUMBER(1));
```

```
-- Update value of new attribute (Season_ID)

UPDATE MON_EQUIP_HIRE_FACT_TEMP_V1
SET Season_ID = 1
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '09' AND
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '11';

UPDATE MON_EQUIP_HIRE_FACT_TEMP_V1
SET Season_ID = 2
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '12' OR
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '02';

UPDATE MON_EQUIP_HIRE_FACT_TEMP_V1
SET Season_ID = 3
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '03' AND
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '05';

UPDATE MON_EQUIP_HIRE_FACT_TEMP_V1
SET Season_ID = 4
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '06' AND
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '08';
```

	TIME_ID	CUSTOMER_TYPE_ID	COMPANY_BRANCH_NAME	CATEGORY_ID	TOTAL_HIRE_PRICE	HIRE_ID	QUANTITY	SEASON_ID
1	201805	1	Hughesdale	12	720	1	3	3
2	201805	2	Geelong	5	3960	2	2	3
3	201805	2	Clayton	10	150	3	1	3
4	201805	2	Cheltenham	3	2160	4	1	3
5	201805	2	Docklands	3	2400	5	2	3
6	201805	1	Richmond	5	4000	6	2	3
7	201805	1	Caulfield	7	1680	7	3	3
8	201805	2	Eltham	12	720	8	3	3
9	201805	2	Clayton	11	170	9	2	3
10	201805	1	Docklands	8	200	10	2	3
11	201806	1	Toorak	6	1800	11	2	4
12	201806	1	Clayton	2	360	12	2	4
13	201806	1	Caulfield	6	450	13	3	4
14	201806	1	Caulfield	8	315	14	3	4
15	201806	2	Docklands	1	360	15	1	4

```
-- MON_EQUIP_HIRE Fact Table V1

CREATE TABLE MON_EQUIP_HIRE_FACT_V1 AS
    SELECT
        Time_ID,
        Season_ID,
        Customer_Type_ID,
        Company_Branch_Name,
        Category_ID,
        SUM(Total_Hire_Price) AS Total_Hire_Revenue,
        COUNT(Hire_ID) AS Total_Number_of_Hire,
        SUM(Quantity) AS Total_Equipment_Hired
    FROM
        MON_EQUIP_HIRE_FACT_TEMP_V1
    GROUP BY
        Time_ID,
        Season_ID,
        Customer_Type_ID,
        Company_Branch_Name,
        Category_ID;
```

	TIME_ID	SEASON_ID	CUSTOMER_TYPE_ID	COMPANY_BRANCH_NAME	CATEGORY_ID	TOTAL_HIRE_REVENUE	TOTAL_NUMBER_OF_HIRE	TOTAL_EQUIPMENT_HIRED
1	201805	3	2	Clayton	10	150	1	1
2	201805	3	1	Richmond	5	4000	1	2
3	201805	3	2	Clayton	11	170	1	2
4	201805	3	1	Docklands	8	200	1	2
5	201807	4	2	Caulfield	8	500	1	2
6	201808	4	2	Docklands	12	4800	1	2
7	201808	4	2	Chadstone	4	960	1	3
8	201809	1	1	Caulfield	2	105	1	3
9	201810	1	2	Clayton	9	5040	1	3
10	201812	2	2	Caulfield	9	40	1	1
11	201812	2	2	Clayton	6	3200	1	2
12	201901	2	2	Eltham	5	750	1	3
13	201903	3	1	Clayton	5	330	1	2
14	201904	3	1	Clayton	1	1600	1	1
15	201905	3	1	Dandenong	13	1600	1	2

Star Schema Version 2

```
-------------------------------------------------------------------------------
-- SALES_DIM
-------------------------------------------------------------------------------
CREATE TABLE SALES_DIM AS
    SELECT
        SALES_ID,
        SALES_DATE,
        QUANTITY,
        UNIT_SALES_PRICE,
        TOTAL_SALES_PRICE
    FROM SALES
    ORDER BY SALES_ID;
```

	SALES_ID	SALES_DATE	QUANTITY	UNIT_SALES_PRICE	TOTAL_SALES_PRICE
1	1	09/04/2018	3	11000	33000
2	2	17/04/2018	2	9000	18000
3	3	10/05/2018	2	41600	83200
4	4	05/06/2018	2	9000	18000
5	5	06/07/2018	4	65000	260000
6	6	07/07/2018	1	63000	63000
7	7	09/07/2018	1	19200	19200
8	8	09/07/2018	4	84500	338000
9	9	10/07/2018	3	54000	162000
10	10	10/07/2018	4	54000	216000
11	11	11/07/2018	1	35000	35000
12	12	11/07/2018	2	40000	80000
13	13	12/07/2018	3	20000	60000
14	14	19/07/2018	3	8800	26400
15	15	31/07/2018	2	5600	11200

```
-------------------------------------------------------------------------------
-- CUSTOMER_DIM
-------------------------------------------------------------------------------

CREATE TABLE CUSTOMER_DIM AS
    SELECT
        C.CUSTOMER_ID,
        C.NAME,
        C.GENDER,
        C.PHONE,
        C.EMAIL,
        CT.DESCRIPTION AS CUSTOMER_TYPE_DESCRIPTION
    FROM CUSTOMER C, MONEQUIP.CUSTOMER_TYPE CT
    WHERE C.CUSTOMER_TYPE_ID = CT.CUSTOMER_TYPE_ID
    ORDER BY CUSTOMER_ID;
```

	CUSTOMER_ID	NAME	GENDER	PHONE	EMAIL	CUSTOMER_TYPE_DESCRIPTION
1	1	Regina Isaacson	Female	601 627 5878	risaacson0@tamu.edu	Individual
2	2	Jaime Whate	Male	318 998 0883	jwhatel@ucoz.ru	Business
3	3	Thaine Hirche	Male	276 571 7986	thirche2@reference.com	Individual
4	4	Deirdre Reddington	Female	585 183 1946	dreddington3@cloudflare.com	Individual
5	5	Domenic Kirrens	Male	798 585 9171	dkirrens4@virginia.edu	Individual
6	6	Kerk Petera	Male	856 940 2206	kpetera5@fastcompany.com	Individual
7	7	Pammie Futter	Female	891 227 4556	pfutter6@woothemes.com	Individual
8	8	Blaire Christopherson	Female	872 144 2174	bchristopherson7@photobucket.com	Business
9	9	Gaye Kemmis	Female	746 484 4734	gkemmis8@vimeo.com	Individual
10	10	Cherise Alessandretti	Female	501 251 3910	calessandretti9@auda.org.au	Business
11	11	Kimmi Deeks	Female	128 972 8249	kdeeksa@who.int	Business
12	12	Leticia Braiden	Female	367 506 7975	lbraidenb@dailymail.co.uk	Individual
13	13	Orel Greschik	Female	866 848 2152	ogreschikc@facebook.com	Business
14	14	Saw Gulliver	Male	387 132 6717	sgulliverd@paypal.com	Individual
15	15	Francesco Della	Male	447 322 8294	fdellae@icio.us	Individual

```
-------------------------------------------------------------------------------
-- STAFF_DIM
-------------------------------------------------------------------------------

CREATE TABLE STAFF_DIM AS
    SELECT
        STAFF_ID,
        FIRST_NAME,
        LAST_NAME,
        GENDER,
        PHONE,
        EMAIL,
        COMPANY_BRANCH
    FROM MONEQUIP.STAFF
    ORDER BY STAFF_ID;
```

	STAFF_ID	FIRST_NAME	LAST_NAME	GENDER	PHONE	EMAIL	COMPANY_BRANCH
1	1	Carleen	Razzell	Female	323 545 5764	carleen.razzell@monequip.com.au	Caulfield
2	2	Ailee	Paxeford	Female	987 455 1555	ailee.paxeford@monequip.com.au	Hughesdale
3	3	Elissa	Danovich	Female	286 378 7209	elissa.danovich@monequip.com.au	Clayton
4	4	Sonnnie	Chestnutt	Female	245 231 1339	sonnnie.chestnutt@monequip.com.au	Toorak
5	5	Mariska	Holtum	Female	262 960 8943	mariska.holtum@monequip.com.au	Clayton
6	6	Egbert	Earl	Male	290 507 8778	egbert.earl@monequip.com.au	Eltham
7	7	Marylinda	Chanders	Female	398 888 9947	marylinda.chanders@monequip.com.au	Chadstone
8	8	Marcella	Diggons	Female	395 748 7317	marcella.diggons@monequip.com.au	Docklands
9	9	Bethina	Gateman	Female	891 703 6967	bethina.gateman@monequip.com.au	Parkville
10	10	Felecia	Stobbart	Female	735 724 1655	felecia.stobbart@monequip.com.au	Caulfield
11	11	Gratia	MacAlinden	Female	986 594 1206	gratia.macalinden@monequip.com.au	Pakenham
12	12	Arleen	Addison	Female	827 178 5759	arleen.addison@monequip.com.au	Clayton
13	13	Ike	Chadbourne	Male	927 633 9154	ike.chadbourne@monequip.com.au	Pakenham
14	14	Dawn	Vaadeland	Female	643 505 2513	dawn.vaadeland@monequip.com.au	Dandenong
15	15	Fergus	Colvill	Male	477 108 6942	fergus.colvill@monequip.com.au	Richmond

```
-------------------------------------------------------------------------------
-- EQUIPMENT_DIM
-------------------------------------------------------------------------------

CREATE TABLE EQUIPMENT_DIM AS
    SELECT
        E.EQUIPMENT_ID,
        E.EQUIPMENT_NAME,
        E.EQUIPMENT_PRICE,
        E.MANUFACTURE_YEAR,
        E.MANUFACTURER,
        C.CATEGORY_DESCRIPTION
    FROM EQUIPMENT E, CATEGORY C
    WHERE E.CATEGORY_ID = C.CATEGORY_ID
    ORDER BY EQUIPMENT_ID;
```

	EQUIPMENT_ID	EQUIPMENT_NAME	EQUIPMENT_PRICE	MANUFACTURE_YEAR	MANUFACTURER	CATEGORY_DESCRIPTION
1	1	SCISSORLIFT 3.0M (10FT) MANUAL	27000	2001	Kenworth	Access
2	2	MANLIFT 4.75M (15FT) SELF PROPELLED	15750	2000	Hitachi	Access
3	3	SCISSORLIFT 5.8M (19FT) ELECTRIC	10800	2008	Volvo	Access
4	4	SCISSORLIFT 5.8M (19FT) TRACKED BI-LEVELLING NARROW	20540	2016	Kobelco	Access
5	5	SCISSORLIFT 7.7M (26FT) ELECTRIC NARROW	15600	2007	Yanmar	Access
6	6	MOBILE HYDRAULIC PLATFORM 19M TRACKED	35750	2013	Caterpillar	Access
7	7	MOBILE HYDRAULIC PLATFORM 9M TRACKED	22200	2013	Isuzu	Access
8	8	MOBILE HYDRAULIC PLATFORM 12M	25200	2005	Volvo	Access
9	9	BOOMLIFT 18M (60FT) DIESEL/ELECTRIC 4WD	21600	2011	Caterpillar	Access
10	10	BOOMLIFT 9M (30FT) ELECTRIC	20160	2014	Kubota	Access
11	11	BOOMLIFT 10.2M (34FT) DIESEL/ELECTRIC 4WD	14400	2000	Hitachi	Access
12	12	MANLIFT 8M SELF PROPELLED	14560	2009	Komatsu	Access
13	13	MANLIFT 6M (19FT) SELF PROPELLED	30000	2010	Volvo	Access
14	14	AIR COMPRESSOR 2.1 CFM 12V	780	2015	Case	Air Compressor
15	15	AIR COMPRESSOR 2.1 CFM (ELECTRIC)	3920	2005	Hitachi	Air Compressor

```
-------------------------------------------------------------------------------
-- HIRE_DIM
-------------------------------------------------------------------------------
CREATE TABLE HIRE_DIM AS
    SELECT
        HIRE_ID,
        START_DATE,
        END_DATE,
        QUANTITY,
        UNIT_HIRE_PRICE,
        TOTAL_HIRE_PRICE
    FROM HIRE
    ORDER BY HIRE_ID;
```

	HIRE_ID	START_DATE	END_DATE	QUANTITY	UNIT_HIRE_PRICE	TOTAL_HIRE_PRICE
1	1	11/05/2018	14/05/2018	3	80	720
2	2	17/05/2018	20/05/2018	2	660	3960
3	3	18/05/2018	19/05/2018	1	150	150
4	4	21/05/2018	25/05/2018	1	540	2160
5	5	21/05/2018	23/05/2018	2	600	2400
6	6	22/05/2018	26/05/2018	2	500	4000
7	7	24/05/2018	28/05/2018	3	140	1680
8	8	25/05/2018	28/05/2018	3	80	720
9	9	28/05/2018	28/05/2018	2	170	170
10	10	29/05/2018	29/05/2018	2	200	200
11	11	12/06/2018	14/06/2018	2	450	1800
12	12	15/06/2018	15/06/2018	2	360	360
13	13	24/06/2018	25/06/2018	3	150	450
14	14	25/06/2018	25/06/2018	3	210	315
15	15	30/06/2018	01/07/2018	1	360	360

```
-------------------------------------------------------------------------------
-- MON_EQUIP_SALES Temporary Fact Table V2
-------------------------------------------------------------------------------
CREATE TABLE MON_EQUIP_SALES_FACT_TEMP_V2 AS
    SELECT
        S.SALES_ID,
        C.CUSTOMER_ID,
        F.STAFF_ID,
        E.EQUIPMENT_ID,
        S.TOTAL_SALES_PRICE,
        S.QUANTITY
    FROM
        SALES S,
        CUSTOMER  C,
        MONEQUIP.STAFF F,
        EQUIPMENT E
    WHERE
        S.CUSTOMER_ID = C.CUSTOMER_ID AND
        S.STAFF_ID = F.STAFF_ID AND
        S.EQUIPMENT_ID = E.EQUIPMENT_ID;
```

	SALES_ID	CUSTOMER_ID	STAFF_ID	EQUIPMENT_ID	TOTAL_SALES_PRICE	QUANTITY
1	131	49	42	2	94500	3
2	34	97	4	2	63000	2
3	79	99	46	4	154050	3
4	57	84	21	5	156000	4
5	30	64	8	6	214500	3
6	76	107	17	7	88800	2
7	61	42	31	15	29400	3
8	113	5	10	18	130000	4
9	85	126	11	19	21600	1
10	121	95	7	20	91000	2
11	150	2	37	20	182000	4
12	151	2	37	20	182000	4
13	84	120	16	20	91000	2
14	71	35	13	22	90000	2
15	53	121	48	24	150000	2

```
-------------------------------------------------------------------------------
-- MON_EQUIP_SALES Fact Table V2
-------------------------------------------------------------------------------
CREATE TABLE MON_EQUIP_SALES_FACT_V2 AS
    SELECT
        SALES_ID,
        CUSTOMER_ID,
        STAFF_ID,
        EQUIPMENT_ID,
        SUM(TOTAL_SALES_PRICE) AS Total_Sales_Revenue,
        SUM(QUANTITY) AS Total_Equipment_Sold
    FROM
        MON_EQUIP_SALES_FACT_TEMP_V2
    GROUP BY
        SALES_ID,
        CUSTOMER_ID,
        STAFF_ID,
        EQUIPMENT_ID;
```

	SALES_ID	CUSTOMER_ID	STAFF_ID	EQUIPMENT_ID	TOTAL_SALES_REVENUE	TOTAL_EQUIPMENT_SOLD
1	150	2	37	20	182000	4
2	17	72	21	27	18000	1
3	75	50	1	34	18000	1
4	58	119	19	41	10000	1
5	108	137	48	42	24000	3
6	119	145	48	59	240000	4
7	51	64	22	67	120000	4
8	41	56	28	70	60800	2
9	87	149	31	84	112000	4
10	82	134	7	104	16800	3
11	45	67	26	118	46800	3
12	2	72	29	128	18000	2
13	115	31	25	138	21600	3
14	20	66	36	145	14000	1
15	63	70	14	152	18000	1

```
-------------------------------------------------------------------------------
-- MON_EQUIP_HIRE Temporary Fact Table V2
-------------------------------------------------------------------------------
CREATE TABLE MON_EQUIP_HIRE_FACT_TEMP_V2 AS
    SELECT
        H.HIRE_ID,
        C.CUSTOMER_ID,
        F.STAFF_ID,
        E.EQUIPMENT_ID,
        H.TOTAL_HIRE_PRICE,
        H.QUANTITY
    FROM
        HIRE H,
        CUSTOMER  C,
        MONEQUIP.STAFF F,
        EQUIPMENT E
    WHERE
        H.CUSTOMER_ID = C.CUSTOMER_ID AND
        H.STAFF_ID = F.STAFF_ID AND
        H.EQUIPMENT_ID = E.EQUIPMENT_ID;
```

	HIRE_ID	CUSTOMER_ID	STAFF_ID	EQUIPMENT_ID	TOTAL_HIRE_PRICE	QUANTITY
1	1	77	2	135	720	3
2	2	70	38	49	3960	2
3	3	58	17	117	150	1
4	4	124	41	36	2160	1
5	5	87	35	37	2400	2
6	6	15	31	53	4000	2
7	7	77	10	73	1680	3
8	8	140	40	135	720	3
9	9	8	46	127	170	2
10	10	5	28	86	200	2
11	11	145	21	71	1800	2
12	12	74	43	23	360	2
13	13	114	10	61	450	3
14	14	110	44	85	315	3
15	15	47	8	9	360	1

```
-------------------------------------------------------------------------------
-- MON_EQUIP_HIRE Fact Table V2
-------------------------------------------------------------------------------
CREATE TABLE MON_EQUIP_HIRE_FACT_V2 AS
    SELECT
        HIRE_ID,
        CUSTOMER_ID,
        STAFF_ID,
        EQUIPMENT_ID,
        SUM(TOTAL_HIRE_PRICE) AS Total_Hire_Revenue,
        SUM(QUANTITY) AS Total_Equipment_Hired
    FROM
        MON_EQUIP_HIRE_FACT_TEMP_V2
    GROUP BY
        HIRE_ID,
        CUSTOMER_ID,
        STAFF_ID,
        EQUIPMENT_ID;
```

	HIRE_ID	CUSTOMER_ID	STAFF_ID	EQUIPMENT_ID	TOTAL_HIRE_REVENUE	TOTAL_EQUIPMENT_HIRED
1	3	58	17	117	150	1
2	4	124	41	36	2160	1
3	7	77	10	73	1680	3
4	8	140	40	135	720	3
5	30	79	34	64	110	1
6	41	100	34	45	960	3
7	62	99	44	123	160	2
8	65	32	2	23	3200	2
9	71	25	29	133	4800	2
10	77	24	10	101	40	1
11	85	31	9	137	60	3
12	91	102	27	148	1440	3
13	109	88	32	112	2640	2
14	112	89	17	13	1600	1
15	118	45	47	17	540	3

SQL STATEMENTS FOR BOTH STAR SCHEMAS

Star Schema Version 1

```
-- Newly created tables during data cleaning phase:
-- CUSTOMER, HIRE, EQUIPMENT, SALES, CATEGORY


------------------------------------------------------------------------
-- DROP TABLE SEGMENT
------------------------------------------------------------------------
drop table sales_price_scale_dim;
drop table season_dim;
drop table company_branch_dim;
drop table time_dim_temp;
drop table time_dim;
drop table customer_type_dim;
drop table category_dim;
drop table MON_EQUIP_SALES_FACT_TEMP_V1;
drop table MON_EQUIP_SALES_FACT_V1;
drop table MON_EQUIP_HIRE_FACT_TEMP_V1;
drop table MON_EQUIP_HIRE_FACT_V1;


-- SALES_PRICE_SCALE Dimension

CREATE TABLE SALES_PRICE_SCALE_DIM
   (
      Sales_Price_Scale_ID NUMBER(5),
      Description VARCHAR2(50),
      Min_Price NUMBER(10),
      Max_Price NUMBER(10)
   );

-- Insert records into SALES_PRICE_SCALE Dimension

INSERT INTO SALES_PRICE_SCALE_DIM VALUES (1, 'Low sales', 0, 4999);
INSERT INTO SALES_PRICE_SCALE_DIM VALUES (2, 'Medium sales', 5000, 10000);
INSERT INTO SALES_PRICE_SCALE_DIM VALUES (3, 'High sales', 10001, 999999);

-- SEASON Dimension

CREATE TABLE SEASON_DIM
   (
      Season_ID NUMBER(1),
      Description VARCHAR2(30),
      Start_Month VARCHAR2(2),
      End_Month VARCHAR2(2)
   );

-- Insert records into SEASON Dimension

INSERT INTO SEASON_DIM VALUES (1, 'Spring', '09', '11');
INSERT INTO SEASON_DIM VALUES (2, 'Summer', '12', '02');
INSERT INTO SEASON_DIM VALUES (3, 'Autumn', '03', '05');
INSERT INTO SEASON_DIM VALUES (4, 'Winter', '06', '08');
```

```
-- COMPANY_BRANCH Dimension

CREATE TABLE COMPANY_BRANCH_DIM AS
   SELECT DISTINCT Company_Branch AS Company_Branch_Name
   FROM MONEQUIP.STAFF
   ORDER BY Company_Branch_Name;


-- Temporary TIME Dimension

CREATE TABLE TIME_DIM_TEMP
(
   TIme_ID VARCHAR2(6)
);

-- Insert records into temporary TIME Dimension

INSERT INTO TIME_DIM_TEMP
SELECT DISTINCT TO_CHAR(Sales_Date, 'YYYYMM') AS Time_ID
FROM SALES;

INSERT INTO TIME_DIM_TEMP
SELECT DISTINCT TO_CHAR(Start_Date, 'YYYYMM') AS Time_ID
FROM HIRE;

-- TIME Dimension

CREATE TABLE TIME_DIM
(
   Time_ID VARCHAR2(6),
   Month VARCHAR2(2),
   Year VARCHAR2(4)
);

-- Insert records into TIME Dimension

INSERT INTO TIME_DIM
SELECT
   DISTINCT Time_ID,
   TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM'),
   TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'YYYY')
FROM TIME_DIM_TEMP;

-- CUSTOMER_TYPE Dimension

CREATE TABLE CUSTOMER_TYPE_DIM AS
   SELECT *
   FROM MONEQUIP.CUSTOMER_TYPE
   ORDER BY CUSTOMER_TYPE_ID;

-- CATEGORY Dimension

CREATE TABLE CATEGORY_DIM AS
   SELECT
      CATEGORY_ID,
      CATEGORY_DESCRIPTION AS Description
```

```
   FROM CATEGORY
   ORDER BY CATEGORY_ID;

-- MON_EQUIP_SALES Temporary Fact Table V1

CREATE TABLE MON_EQUIP_SALES_FACT_TEMP_V1 AS
   SELECT
      TO_CHAR(S.Sales_Date, 'YYYYMM') AS Time_ID,
      C.Customer_Type_ID,
      F.Company_Branch AS Company_Branch_Name,
      Y.Category_ID,
      S.Total_Sales_Price,
      S.Sales_ID,
      S.Quantity
   FROM
      SALES S,
      CUSTOMER C,
      MONEQUIP.STAFF F,
      EQUIPMENT E,
      CATEGORY Y
   WHERE
      S.Customer_ID = C.Customer_ID AND
      S.Staff_ID = F.Staff_ID AND
      S.Equipment_ID = E.Equipment_ID AND
      E.Category_ID = Y.Category_ID;

-- Add new attributes

ALTER TABLE MON_EQUIP_SALES_FACT_TEMP_V1
ADD (Season_ID NUMBER(1));

ALTER TABLE MON_EQUIP_SALES_FACT_TEMP_V1
ADD (Sales_Price_Scale_ID NUMBER(10));

-- Update value of new attribute (Season_ID)

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Season_ID = 1
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '09' AND
   TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '11';

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Season_ID = 2
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '12' OR
   TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '02';

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Season_ID = 3
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '03' AND
   TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '05';

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Season_ID = 4
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '06' AND
   TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '08';

-- Update value of new attribute (Sales_Price_Scale_ID)
```

```
UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Sales_Price_Scale_ID = 1
WHERE Total_Sales_Price BETWEEN 0 AND 4999;

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Sales_Price_Scale_ID = 2
WHERE Total_Sales_Price BETWEEN 5000 AND 10000;

UPDATE MON_EQUIP_SALES_FACT_TEMP_V1
SET Sales_Price_Scale_ID = 3
WHERE Total_Sales_Price BETWEEN 10001 AND 999999;

-- MON_EQUIP_SALES Fact Table V1

CREATE TABLE MON_EQUIP_SALES_FACT_V1 AS
   SELECT
      Time_ID,
      Season_ID,
      Customer_Type_ID,
      Company_Branch_Name,
      Category_ID,
      Sales_Price_Scale_ID,
      SUM(Total_Sales_Price) AS Total_Sales_Revenue,
      COUNT(Sales_ID) AS Total_Number_of_Sales,
      SUM(Quantity) AS Total_Equipment_Sold
   FROM
      MON_EQUIP_SALES_FACT_TEMP_V1
   GROUP BY
      Time_ID,
      Season_ID,
      Customer_Type_ID,
      Company_Branch_Name,
      Category_ID,
      Sales_Price_Scale_ID;

-- MON_EQUIP_HIRE Temporary Fact Table V1

CREATE TABLE MON_EQUIP_HIRE_FACT_TEMP_V1 AS
   SELECT
      TO_CHAR(H.Start_Date, 'YYYYMM') AS Time_ID,
      C.Customer_Type_ID,
      F.Company_Branch AS Company_Branch_Name,
      Y.Category_ID,
      H.Total_Hire_Price,
      H.Hire_ID,
      H.Quantity
   FROM
      HIRE H,
      CUSTOMER C,
      MONEQUIP.STAFF F,
      EQUIPMENT E,
      CATEGORY Y
   WHERE
      H.Customer_ID = C.Customer_ID AND
      H.Staff_ID = F.Staff_ID AND
      H.Equipment_ID = E.Equipment_ID AND
```

```
        E.Category_ID = Y.Category_ID;

-- Add new attributes

ALTER TABLE MON_EQUIP_HIRE_FACT_TEMP_V1
ADD (Season_ID NUMBER(1));

-- Update value of new attribute (Season_ID)

UPDATE MON_EQUIP_HIRE_FACT_TEMP_V1
SET Season_ID = 1
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '09' AND
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '11';

UPDATE MON_EQUIP_HIRE_FACT_TEMP_V1
SET Season_ID = 2
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '12' OR
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '02';

UPDATE MON_EQUIP_HIRE_FACT_TEMP_V1
SET Season_ID = 3
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '03' AND
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '05';

UPDATE MON_EQUIP_HIRE_FACT_TEMP_V1
SET Season_ID = 4
WHERE TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') >= '06' AND
    TO_CHAR(TO_DATE(Time_ID, 'YYYYMM'), 'MM') <= '08';

-- MON_EQUIP_HIRE Fact Table V1

CREATE TABLE MON_EQUIP_HIRE_FACT_V1 AS
    SELECT
        Time_ID,
        Season_ID,
        Customer_Type_ID,
        Company_Branch_Name,
        Category_ID,
        SUM(Total_Hire_Price) AS Total_Hire_Revenue,
        COUNT(Hire_ID) AS Total_Number_of_Hire,
        SUM(Quantity) AS Total_Equipment_Hired
    FROM
        MON_EQUIP_HIRE_FACT_TEMP_V1
    GROUP BY
        Time_ID,
        Season_ID,
        Customer_Type_ID,
        Company_Branch_Name,
        Category_ID;
```

Star Schema Version 2

```
-- Newly created tables during data cleaning phase:
-- CUSTOMER, HIRE, EQUIPMENT, SALES, CATEGORY


-------------------------------------------------------------------------
-- DROP TABLE SEGMENT
-------------------------------------------------------------------------
drop table sales_dim;
drop table customer_dim;
drop table staff_dim;
drop table equipment_dim;
drop table hire_dim;
drop table MON_EQUIP_SALES_FACT_TEMP_V2;
drop table MON_EQUIP_SALES_FACT_V2;
drop table MON_EQUIP_HIRE_FACT_TEMP_V2;
drop table MON_EQUIP_HIRE_FACT_V2;


-------------------------------------------------------------------------
-- SALES_DIM, CUSTOMER_DIM, STAFF_DIM, EQUIPMENT_DIM, HIRE_DIM
-------------------------------------------------------------------------
CREATE TABLE SALES_DIM AS
   SELECT
      SALES_ID,
      SALES_DATE,
      QUANTITY,
      UNIT_SALES_PRICE,
      TOTAL_SALES_PRICE
   FROM SALES
   ORDER BY SALES_ID;

CREATE TABLE CUSTOMER_DIM AS
   SELECT
      C.CUSTOMER_ID,
      C.NAME,
      C.GENDER,
      C.PHONE,
      C.EMAIL,
      CT.DESCRIPTION AS CUSTOMER_TYPE_DESCRIPTION
   FROM CUSTOMER C, MONEQUIP.CUSTOMER_TYPE CT
   WHERE C.CUSTOMER_TYPE_ID = CT.CUSTOMER_TYPE_ID
   ORDER BY CUSTOMER_ID;

CREATE TABLE STAFF_DIM AS
   SELECT
      STAFF_ID,
      FIRST_NAME,
      LAST_NAME,
      GENDER,
      PHONE,
      EMAIL,
      COMPANY_BRANCH
   FROM MONEQUIP.STAFF
   ORDER BY STAFF_ID;
```

```
CREATE TABLE EQUIPMENT_DIM AS
   SELECT
      E.EQUIPMENT_ID,
      E.EQUIPMENT_NAME,
      E.EQUIPMENT_PRICE,
      E.MANUFACTURE_YEAR,
      E.MANUFACTURER,
      C.CATEGORY_DESCRIPTION
   FROM EQUIPMENT E, CATEGORY C
   WHERE E.CATEGORY_ID = C.CATEGORY_ID
   ORDER BY EQUIPMENT_ID;

CREATE TABLE HIRE_DIM AS
   SELECT
      HIRE_ID,
      START_DATE,
      END_DATE,
      QUANTITY,
      UNIT_HIRE_PRICE,
      TOTAL_HIRE_PRICE
   FROM HIRE
   ORDER BY HIRE_ID;


-------------------------------------------------------------------------
-- MON_EQUIP_SALES Temporary Fact Table V2
-------------------------------------------------------------------------
CREATE TABLE MON_EQUIP_SALES_FACT_TEMP_V2 AS
   SELECT
      S.SALES_ID,
      C.CUSTOMER_ID,
      F.STAFF_ID,
      E.EQUIPMENT_ID,
      S.TOTAL_SALES_PRICE,
      S.QUANTITY
   FROM
      SALES S,
      CUSTOMER  C,
      MONEQUIP.STAFF F,
      EQUIPMENT E
   WHERE
      S.CUSTOMER_ID = C.CUSTOMER_ID AND
      S.STAFF_ID = F.STAFF_ID AND
      S.EQUIPMENT_ID = E.EQUIPMENT_ID;


-------------------------------------------------------------------------
-- MON_EQUIP_SALES Fact Table V2
-------------------------------------------------------------------------
CREATE TABLE MON_EQUIP_SALES_FACT_V2 AS
   SELECT
      SALES_ID,
      CUSTOMER_ID,
      STAFF_ID,
      EQUIPMENT_ID,
      SUM(TOTAL_SALES_PRICE) AS Total_Sales_Revenue,
      SUM(QUANTITY) AS Total_Equipment_Sold
```

```
   FROM
      MON_EQUIP_SALES_FACT_TEMP_V2
   GROUP BY
      SALES_ID,
      CUSTOMER_ID,
      STAFF_ID,
      EQUIPMENT_ID;


-------------------------------------------------------------------------
-- MON_EQUIP_HIRE Temporary Fact Table V2
-------------------------------------------------------------------------
CREATE TABLE MON_EQUIP_HIRE_FACT_TEMP_V2 AS
   SELECT
      H.HIRE_ID,
      C.CUSTOMER_ID,
      F.STAFF_ID,
      E.EQUIPMENT_ID,
      H.TOTAL_HIRE_PRICE,
      H.QUANTITY
   FROM
      HIRE H,
      CUSTOMER  C,
      MONEQUIP.STAFF F,
      EQUIPMENT E
   WHERE
      H.CUSTOMER_ID = C.CUSTOMER_ID AND
      H.STAFF_ID = F.STAFF_ID AND
      H.EQUIPMENT_ID = E.EQUIPMENT_ID;

-------------------------------------------------------------------------
-- MON_EQUIP_HIRE Fact Table V2
-------------------------------------------------------------------------
CREATE TABLE MON_EQUIP_HIRE_FACT_V2 AS
   SELECT
      HIRE_ID,
      CUSTOMER_ID,
      STAFF_ID,
      EQUIPMENT_ID,
      SUM(TOTAL_HIRE_PRICE) AS Total_Hire_Revenue,
      SUM(QUANTITY) AS Total_Equipment_Hired
   FROM
      MON_EQUIP_HIRE_FACT_TEMP_V2
   GROUP BY
      HIRE_ID,
      CUSTOMER_ID,
      STAFF_ID,
      EQUIPMENT_ID;
```

CONCLUSION

In conclusion, we have developed a data warehouse and cleaned the existing database from inconsistencies to enhance Monash Equipment Centre's (MonEquip) data management and business intelligence capabilities. By addressing errors in the existing data, our team has fortified the integrity of MonEquip's information resources. Leveraging multiple methods for data cleaning and exploration, we have not only rectified inaccuracies but also unlocked valuable insights embedded within the dataset. In addition, the creation of two distinct star schemas, finely tuned to different aggregations, signifies a tailored approach aimed at providing deeper insights and operational efficiency. The implemented data warehouse, coupled with optimised star schemas, will empower informed decision-making and strategic foresight for MonEquip’s continued success.