SIT772 2022 T1

Assignment 1b

YANJUN CHEN
216327327

1.1

```sql
--------------------------
-- Task 1.1: [1 Marks]
-- Write one SQL query to list the region names and the number of countries for each region from the above database.
--------------------------
SELECT regions.region_name, COUNT(countries.region_id) AS countries_count
    FROM regions
LEFT JOIN countries
    ON regions.region_id = countries.region_id
GROUP BY regions.region_name
;
```

REGION_NAME	COUNTRIES_COUNT
Europe	8
Americas	5
Asia	6
Middle East and Africa	6

1.2

```sql
--------------------------
-- Task 1.2: [1 Marks]
-- Write one SQL query to find all customers who have made orders before 2016. List must include the customer ID, customer name, and ordered by their ID values in descending.
--------------------------
SELECT customers.customer_id, customers.name, orders.order_date
    FROM customers
INNER JOIN orders
    ON customers.customer_id = orders.customer_id
WHERE orders.order_date < to_date('1-JAN-16','DD-MON-RR')
ORDER BY customers.customer_id DESC
;
```

```
CUSTOMER_ID
-----------
NAME
--------------------
ORDER_DAT
---------
        70
PPG Industries
29-JUN-15

        56
Nucor
03-MAY-15

CUSTOMER_ID
-----------
NAME
--------------------
ORDER_DAT
---------

        53
NGL Energy Partners
31-MAY-15

        47
General Mills

CUSTOMER_ID
-----------
NAME
--------------------
ORDER_DAT
---------
02-MAY-15

        44
Jabil Circuit
28-OCT-15

         8
```

```
CUSTOMER_ID
-----------
NAME
--------------------
ORDER_DAT
---------
International Paper
09-APR-15

         8
International Paper
28-OCT-15


CUSTOMER_ID
-----------
NAME
--------------------
ORDER_DAT
---------
         7
Alcoa
27-OCT-15

         6
Community Health Syst
09-APR-15
```

```
CUSTOMER_ID
-----------
NAME
--------------------------
ORDER_DAT
---------

         6
Community Health Systems
21-JUN-13

         4
AbbVie

CUSTOMER_ID
-----------
NAME
--------------------------
ORDER_DAT
---------
26-APR-15

         3
US Foods Holding
01-DEC-15

         3

CUSTOMER_ID
-----------
NAME
--------------------------
ORDER_DAT
---------
US Foods Holding
30-JUN-15

         2
Plains GP Holdings
14-DEC-15


14 rows selected.
```

1.3

```sql
-------------------------
-- Task 1.3: [1 Marks]
-- Write one SQL query to list all customers who have the sequential letters 'co' in the customer name
where the condition 'co' is case-insensitive, i.e., 'CO', 'Co', … can also be retrieved. List must
include the customers' ID, names and ordered by their names in ascending.
-------------------------
SELECT customers.customer_id, customers.name
    FROM customers
WHERE LOWER(customers.name) LIKE '%co%'
ORDER BY customers.name ASC
;
```

```
CUSTOMER_ID
-----------
NAME
------------------------------
          41
AECOM

           7
Alcoa

         105
Amazon.com


CUSTOMER_ID
-----------
NAME
------------------------------
         319
BB&T Corp.

         114
Bank of America Corp.

          66
Bank of New York Mellon Corp.


CUSTOMER_ID
-----------
NAME
------------------------------
         163
Charter Communications

         146
Cisco Systems

         155
Coca-Cola


CUSTOMER_ID
-----------
NAME
------------------------------
         267
Cognizant Technology Solution

          60
Colgate-Palmolive

         128
Comcast
```

CUSTOMER_ID	NAME
6	Community Health Systems
271	Computer Sciences
62	ConAgra Foods
187	ConocoPhillips
265	Consolidated Edison
310	Core-Mark Holding
294	Corning
102	Costco
82	Discover Financial Services
236	Ecolab
261	Guardian Life Ins. Co. of Ar
273	Jacobs Engineering Group
165	Johnson Controls
284	L-3 Communications
235	Lincoln National
83	National Oilwell Varco
56	Nucor
75	Omnicom Group
52	PG&E Corp.
136	PepsiCo
212	Qualcomm
126	State Farm Insurance Cos.
307	State Street Corp.
149	Sysco
201	Travelers Cos.
12	U.S. Bancorp
177	United Continental Holdings
244	Viacom

40 rows selected.

1.4

```sql
--------------------------
-- Task 1.4: [1 Marks]
-- Write one SQL query to list all products' ID, Name and price where the products haven't been purchased by any customer in the database. The list must be ordered by the product price.
--------------------------
SELECT products.product_id, products.product_name, products.list_price
    FROM products
WHERE products.product_id NOT IN (
    SELECT product_id FROM order_items
    -- this could work when customers must make order and pay before they cancel.
)
ORDER BY products.list_price ASC
;
```

```
PRODUCT_ID
----------
PRODUCT_NAME
--------------------------------
LIST_PRICE
----------
        57
Western Digital WD20EZRZ
     67.34

       232
Western Digital WD1003FZEX
     70.89

PRODUCT_ID
----------
PRODUCT_NAME
--------------------------------
LIST_PRICE
----------

        22
Seagate ST3000DM008
     83.61

        26
Samsung MZ-75E500B/AM

PRODUCT_ID
----------
PRODUCT_NAME
--------------------------------
LIST_PRICE
----------
    178.09

       148
MSI Z270 XPOWER GAMING TITANIUM
    282.98

       140

PRODUCT_ID
----------
PRODUCT_NAME
--------------------------------
LIST_PRICE
----------
MSI X99A WORKSTATION
    289.97

       173
ASRock Z270 SuperCarrier
    353.98
```

PRODUCT_ID	PRODUCT_ID
PRODUCT_NAME	PRODUCT_NAME
LIST_PRICE	LIST_PRICE
229	221
Seagate ST10000DM0004	Zotac ZT-P10810C-10P
399.99	759.99
274	283
ASRock E3C224D4M-16RE	G.Skill Trident Z
499.99	760.99
PRODUCT_ID	PRODUCT_ID
PRODUCT_NAME	PRODUCT_NAME
LIST_PRICE	LIST_PRICE
193	223
Asus Z10PE-D8 WS	MSI GeForce GTX 1080 TI Al
561.59	764.98
117	175
G.Skill Ripjaws V Series	NVIDIA VCQM4000-PB
PRODUCT_ID	PRODUCT_ID
PRODUCT_NAME	PRODUCT_NAME
LIST_PRICE	LIST_PRICE
695.99	790
198	197
Intel Core i7-980	G.Skill Trident Z RGB
699.99	799.99
88	112
PRODUCT_ID	PRODUCT_ID
PRODUCT_NAME	PRODUCT_NAME
LIST_PRICE	LIST_PRICE
Gigabyte GV-N98TWF3OC-6GD	Corsair Vengeance Pro
749.99	808.92
277	59
G.Skill Trident Z	Intel Core i7-5960X (OEM/
758.99	977.99

```
PRODUCT_ID
----------
PRODUCT_NAME
----------------------------
LIST_PRICE
----------
        81
Intel Xeon E5-2650 V4
   1099.99

        37
Corsair Dominator Platinum
   1264.99

PRODUCT_ID
----------
PRODUCT_NAME
----------------------------
LIST_PRICE
----------

       161
AMD 100-5056062
   1499.99

       244
Crucial

PRODUCT_ID
----------
PRODUCT_NAME
----------------------------
LIST_PRICE
----------
   1620.99

       212
Intel Xeon E5-2680 V4
   1639.99

       243

PRODUCT_ID
----------
PRODUCT_NAME
----------------------------
LIST_PRICE
----------
Intel Xeon E5-2643 V4 (OEM/
   1708.86

       267
EVGA 12G-P4-1999-KR
   1799.99
```

```
PRODUCT_ID
----------
PRODUCT_NAME
---------------------------------
LIST_PRICE
----------
       162
Intel Xeon E5-2470V2
    1904.7

        46
Intel Xeon E5-2695 V3 (OEM/Tray)
   2431.95

PRODUCT_ID
----------
PRODUCT_NAME
---------------------------------
LIST_PRICE
----------

       228
Intel Xeon E5-2699 V3 (OEM/Tray)
   3410.46

       133
PNY VCQP6000-PB

PRODUCT_ID
----------
PRODUCT_NAME
---------------------------------
LIST_PRICE
----------
   5499.99


32 rows selected.
```

1.5

```
43  --------------------------
44  -- Task 1.5: [1 Marks]
45  -- Write one SQL query to list the employees and the number of orders that each employee processed in
    the database. The output list must include employee ID, name, and the number of orders. The list must
    be sorted by the number of orders in the descending order.
46  --------------------------
47  SELECT employees.employee_id,
48      CONCAT(CONCAT(employees.first_name,' '), employees.last_name) AS name,
49      COUNT(orders.salesman_id) AS order_count
50      FROM employees
51  LEFT JOIN orders
52      ON orders.salesman_id = employees.employee_id
53  GROUP BY employees.employee_id,
54      CONCAT(CONCAT(employees.first_name,' '), employees.last_name)
55  ORDER BY order_count DESC
56  ;
```

EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
62	56	37	105	38
Freya Gomez	Evie Harrison	Ibrahim Alexander	Gracie Gardner	Sonny Russell
13	5	0	0	0
64	54	103	71	94
Florence Freeman	Lily Fisher	Amelie Hudson	Millie Hunter	Maisie Nichol
12	5	0	0	0
EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
55	36	5	79	63
Grace Ellis	Felix Bryant	Nathan Cox	Esme Warren	Phoebe Murray
10	0	0	0	0
59	1	33	44	30
Chloe Cruz	Tommy Bailey	Reggie Simmons	Emily Hamilton	Austin Flores
EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
7	0	0	0	0
60	31	46	35	17
Isabelle Marshall	Ellis Washington	Ava Sullivan	Carter Gonzales	Frederick Pri
7	0	0	0	0
61	21	107	18	11
EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
Daisy Ortiz	Jaxon Ross	Summer Payne	Ollie Bennett	Tyler Ramirez
6	0	0	0	0
57	76	23	74	10
Scarlett Gibson	Layla Mason	Jackson Coleman	Elsie Henry	Ryan Gray
5	0	0	0	0

EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
53	25	28	81
Sophia Reynolds	Ronnie Perry	Aaron Patterson	Lola Ramos
0	0	0	0
9	14	19	49
Mohammad Peterson	Elliot Brooks	Louis Wood	Isabella Cole
0	0	0	0
EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
20	40	72	48
Dexter Barnes	Caleb Diaz	Sofia Hicks	Jessica Woods
0	0	0	0
67	15	58	65
Sienna Simpson	Rory Kelly	Ruby Mcdonald	Alice Wells
EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
0	0	0	0
83	98	68	87
Ivy Burns	Amber Rose	Matilda Stevens	Molly Rice
0	0	0	0
69	77	73	26
EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
Evelyn Tucker	Rosie Morales	Lucy Crawford	Leon Powell
0	0	0	0
16	66	90	96
Alex Sanders	Charlotte Webb	Eliza Black	Hannah Knight
0	0	0	0

EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
32	82	13	84
Jamie Butler	Willow Reyes	Albert Watson	Erin Gordon
0	0	0	0
92	41	100	2
Abigail Palmer	Connor Hayes	Thea Hawkins	Jude Rivera
0	0	0	0
EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
39	12	89	70
Kian Griffin	Elliott James	Jasmine Hunt	Eva Porter
0	0	0	0
6	101	52	4
Gabriel Howard	Annabelle Dunn	Sophie Owens	Louie Richardson
EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
0	0	0	0
51	22	85	3
Poppy Jordan	Liam Henderson	Holly Shaw	Blake Cooper
0	0	0	0
7	95	42	91
EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME	NAME	NAME
ORDER_COUNT	ORDER_COUNT	ORDER_COUNT	ORDER_COUNT
Charles Ward	Eleanor Grant	Amelia Myers	Lilly Daniels
0	0	0	0
78	29	34	80
Maya Kennedy	Roman Hughes	Seth Foster	Elizabeth Dixon
0	0	0	0

EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME
ORDER_COUNT	ORDER_COUNT
24	45
Callum Jenkins	Isla Graham
0	0
106	104
Rose Stephens	Harper Spencer
0	0
EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME
ORDER_COUNT	ORDER_COUNT
50	8
Mia West	Bobby Torres
0	0
27	75
Kai Long	Imogen Boyd
EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME
ORDER_COUNT	ORDER_COUNT
0	0
88	86
Ellie Robertson	Emilia Holmes
0	0
99	93
EMPLOYEE_ID	EMPLOYEE_ID
NAME	NAME
ORDER_COUNT	ORDER_COUNT
Bella Stone	Georgia Mills
0	0
43	97
Olivia Ford	Harriet Ferguson
0	0

EMPLOYEE_ID
NAME
ORDER_COUNT
102
Emma Perkins
0
47
Ella Wallace
0
EMPLOYEE_ID
NAME
ORDER_COUNT

107 rows selected.

1.6

```sql
--------------------------
-- Task 1.6: [1 Marks]
-- Write one SQL query to list all the warehouses and the revenue of each warehouse. Here, given a
product, the revenue of the product is calculated by the sold quantity of the product and its
List_Price. The list must be ordered by the revenue value in the descending. [Reminder: if one
product_ID links to more than one warehouses in the provided database, you can simply count it into its
linked warehouses' revenue.]
--------------------------
SELECT inventories.warehouse_id, SUM(income) AS revenue
    FROM(
        SELECT id, sold_quantity, products.list_price,
        sold_quantity*products.list_price AS income
            FROM(
                SELECT order_items.product_id AS id,
                    SUM(order_items.quantity) AS sold_quantity
                    FROM order_items
                INNER JOIN orders
                    ON order_items.order_id = orders.order_id
                WHERE orders.status != 'Canceled'
                GROUP BY order_items.product_id
                ORDER BY order_items.product_id
            ), products
        WHERE id = products.product_id
        GROUP BY id, sold_quantity, products.list_price
    ), inventories
WHERE id = inventories.product_id
GROUP BY inventories.warehouse_id
ORDER BY revenue DESC
;
```

```
WAREHOUSE_ID    REVENUE
------------ ----------
           6 28371361.7
           8 26337656.2
           9 24164403.7
           2 22071269.8
           4 18446688.3
           5   16066013
           7   14655859
           3 12482606.5
           1    7375654

9 rows selected.
```

2.1

Task 2.1: [2 Marks]
Normalize the table to ensure all generated tables are in 3NF. Present all tables generated from the normalization and **present the results step by step from 1NF to 3NF**. You are required to specify the primary keys in your generated tables.

***All the primary key of following table has been highlighted with yellow.**

1NF

TransactionDate	Product	Price	Amount	Category	PaymentType	CustomerName	Company Contact	Company	Shop	address
21/03/2021	ThinkPad	1,500	1	Electronic	Visa	Kevin Li	111111	IBM	Freeland Choice	20 Avenue, Burwood, VIC3125
21/03/2021	Mouse	$30	2	Electronic	Visa	Kevin Li	111111	IBM	Freeland Choice	20 Avenue, Burwood, VIC3125
21/03/2021	Office Chair	$79	5	Office	Mastercard	Daniel Andrews	222222	Luxo Narelle Mesh	City Bunnings	36 Maple street, Melbourne, VIC3000
22/03/2021	Camera	$100	1	Electronic	Visa	Kevin Li	111111	IBM	Burwood Electronic	20 Avenue, Burwood
22/03/2021	Keyboard	$50	2	Electronic	Visa	Kevin Li	111111	IBM	Burwood Electronic	20 Avenue, Burwood
23/03/2021	ThinkPad	$1,350	2	Electronic	Mastercard	David Andrews	111111	IBM	Burwood Officework	606 Burwood Hwy, Vermont South VIC 3133
24/03/2021	Office Chair	$80	4	Office	Mastercard	Daniel Andrews	222222	Luxo Narelle Mesh	Burwood Bunnings	606-634 Burwood Hwy, Vermont South VIC 3133

2NF

product_id	store_id	Amount	TransactionDate	Product	Price	Category	Shop	address
1	1	1	21/03/2021	ThinkPad	1,500	Electronic	Freeland Choice	20 Avenue, Burwood, VIC3125
2	1	2	21/03/2021	Mouse	$30	Electronic	Freeland Choice	20 Avenue, Burwood, VIC3125
3	2	5	21/03/2021	Office Chair	$79	Office	City Bunnings	36 Maple street, Melbourne, VIC3000
4	3	1	22/03/2021	Camera	$100	Electronic	Burwood Electronic	20 Avenue, Burwood
5	3	2	22/03/2021	Keyboard	$50	Electronic	Burwood Electronic	20 Avenue, Burwood
1	4	2	23/03/2021	ThinkPad	$1,350	Electronic	Burwood Officework	606 Burwood Hwy, Vermont South VIC 3133
3	5	4	24/03/2021	Office Chair	$80	Office	Burwood Bunnings	606-634 Burwood Hwy, Vermont South VIC 3133

customer_id	CustomerName	Company Contact	Company	PaymentType
1	Kevin Li	111111	IBM	Visa
2	David Andrews	111111	IBM	Mastercard
3	Daniel Andrews	222222	Luxo Narelle Mesh	Mastercard

customer_id	product_id
1	1
1	2
3	3
1	4
1	5
2	6
3	7

3NF

order_number	store_id	customer_id	product_id	Amount	Price	TransactionDate
1	1	1	1	1	1500	21/03/2021
2	1	1	2	2	30	21/03/2021
3	2	3	3	5	79	21/03/2021
4	3	1	4	1	100	22/03/2021
5	3	1	5	2	50	22/03/2021
6	4	2	1	2	1350	23/03/2021
7	5	3	3	4	80	24/03/2021

store_id	Store name	address
1	Freeland Choice	20 Avenue, Burwood, VIC3125
2	City Bunnings	36 Maple street, Melbourne, VIC3000
3	Burwood Electronic	20 Avenue, Burwood
4	Burwood Officework	606 Burwood Hwy, Vermont South VIC 3133
5	Burwood Bunnings	606-634 Burwood Hwy, Vermont South VIC 3133

product_id	Product name	Category
1	ThinkPad	Electronic
2	Mouse	Electronic
3	Office Chair	Office
4	Camera	Electronic
5	Keyboard	Electronic

customer_id	Customer Name	PaymentType	Company_id
1	Kevin Li	Visa	1
2	David Andrews	Mastercard	1
3	Daniel Andrews	Mastercard	2

Company_id	Company Contact	Company name
1	111111	IBM
2	222222	Luxo Narelle Mesh

2.2

```
--------------------------
-- Task 2.2: [1 Marks]
-- Write a set of SQL queries to create your normalized tables for building the database schema. Each table must declare the primary keys, foreign keys if applicable.
-- (Marking Rubric: 1 mark if there is no mistake to create the tables or only has one mistake; otherwise, 0 mark will be given.)
--------------------------
CREATE TABLE product_ (
    product_id NUMBER PRIMARY KEY,
    product_name VARCHAR2 (50),
    category VARCHAR2 (50)
);
CREATE TABLE store_ (
    store_id NUMBER PRIMARY KEY,
    store_name VARCHAR2 (50),
    address VARCHAR2 (255)
);
CREATE TABLE company_ (
    company_id NUMBER PRIMARY KEY,
    company_contact VARCHAR2 (50),
    company_name VARCHAR2 (50)
);
CREATE TABLE customer_ (
    customer_id NUMBER PRIMARY KEY,
    customer_name VARCHAR2 (50),
    payment_type VARCHAR2 (50),
    company_id NUMBER,
    CONSTRAINT fk_customer_company
        FOREIGN KEY (company_id)
        REFERENCES company_ (company_id)
        ON DELETE CASCADE
);
```

```
CREATE TABLE order_ (
    order_number NUMBER PRIMARY KEY,
    store_id NUMBER,
    customer_id NUMBER,
    product_id NUMBER,
    amount NUMBER,
    price NUMBER,
    transaction_date DATE,
    CONSTRAINT fk_order_store
        FOREIGN KEY (store_id)
        REFERENCES store_ (store_id)
        ON DELETE CASCADE,
    CONSTRAINT fk_order_customer
        FOREIGN KEY (customer_id)
        REFERENCES customer_ (customer_id)
        ON DELETE CASCADE,
    CONSTRAINT fk_order_product
        FOREIGN KEY (product_id)
        REFERENCES product_ (product_id)
        ON DELETE CASCADE
);
```

```
Table created.


Table created.


Table created.


Table created.


Table created.
```

2.3

```
--------------------------
-- Task 2.3: [1 Marks]
-- Write a set of SQL queries to add data into the database implemented in Task 2.2. The database must include all the provided information. If need, you can add unique identifiers or ids for tables.
--------------------------
ALTER TABLE customer_ DISABLE CONSTRAINT fk_customer_company;
ALTER TABLE order_ DISABLE CONSTRAINT fk_order_store;
ALTER TABLE order_ DISABLE CONSTRAINT fk_order_customer;
ALTER TABLE order_ DISABLE CONSTRAINT fk_order_product;

REM INSERTING into product_ SET DEFINE OFF;
Insert into product_ (product_id, product_name, category)
    values (1,'ThinkPad', 'Electronic');
Insert into product_ (product_id, product_name, category)
    values (2,'Mouse', 'Electronic');
Insert into product_ (product_id, product_name, category)
    values (3,'Office Chair', 'Office');
Insert into product_ (product_id, product_name, category)
    values (4,'Camera', 'Electronic');
Insert into product_ (product_id, product_name, category)
    values (5,'Keyboard', 'Electronic');
SELECT * FROM product_;

REM INSERTING into store_ SET DEFINE OFF;
Insert into store_ (store_id, store_name, address)
    values (1,'Freeland Choice', '20 Avenue, Burwood, VIC3125');
Insert into store_ (store_id, store_name, address)
    values (2,'City Bunnings', '36 Maple street, Melbourne, VIC3000');
Insert into store_ (store_id, store_name, address)
    values (3,'Burwood Electronic', '20 Avenue, Burwood');
Insert into store_ (store_id, store_name, address)
    values (4,'Burwood Officework', '606 Burwood Hwy, Vermont South VIC 3133');
Insert into store_ (store_id, store_name, address)
    values (5,'Burwood Bunnings', '606-634 Burwood Hwy, Vermont South VIC 3133');
SELECT * FROM store_;
```

```
REM INSERTING into company_ SET DEFINE OFF;
Insert into company_ (company_id, company_contact, company_name)
    values (1, 111111, 'IBM');
Insert into company_ (company_id, company_contact, company_name)
    values (2, 222222, 'Luxo Narelle Mesh');
SELECT * FROM company_;

REM INSERTING into customer_ SET DEFINE OFF;
Insert into customer_ (customer_id, customer_name, payment_type, company_id)
    values (1, 'Kevin Li', 'Visa', 1);
Insert into customer_ (customer_id, customer_name, payment_type, company_id)
    values (2, 'David Andrews', 'Mastercard', 1);
Insert into customer_ (customer_id, customer_name, payment_type, company_id)
    values (3, 'Daniel Andrews', 'Mastercard', 2);
SELECT * FROM customer_;

REM INSERTING into order_ SET DEFINE OFF;
Insert into order_ (order_number, store_id, customer_id, product_id, amount, price, transaction_date)
    values (1, 1, 1, 1, 1, 1500, to_date('21/03/2021','DD/MM/RR'));
Insert into order_ (order_number, store_id, customer_id, product_id, amount, price, transaction_date)
    values (2, 1, 1, 2, 2, 30, to_date('21/03/2021','DD/MM/RR'));
Insert into order_ (order_number, store_id, customer_id, product_id, amount, price, transaction_date)
    values (3, 2, 3, 3, 5, 79, to_date('21/03/2021','DD/MM/RR'));
Insert into order_ (order_number, store_id, customer_id, product_id, amount, price, transaction_date)
    values (4, 3, 1, 4, 1, 100, to_date('22/03/2021','DD/MM/RR'));
Insert into order_ (order_number, store_id, customer_id, product_id, amount, price, transaction_date)
    values (5, 3, 1, 5, 2, 50, to_date('22/03/2021','DD/MM/RR'));
Insert into order_ (order_number, store_id, customer_id, product_id, amount, price, transaction_date)
    values (6, 4, 2, 1, 2, 1350, to_date('23/03/2021','DD/MM/RR'));
Insert into order_ (order_number, store_id, customer_id, product_id, amount, price, transaction_date)
    values (7, 5, 3, 3, 4, 80, to_date('24/03/2021','DD/MM/RR'));
SELECT * FROM order_;

ALTER TABLE customer_ ENABLE CONSTRAINT fk_customer_company;
ALTER TABLE order_ ENABLE CONSTRAINT fk_order_store;
ALTER TABLE order_ ENABLE CONSTRAINT fk_order_customer;
ALTER TABLE order_ ENABLE CONSTRAINT fk_order_product;
```

```
Table altered.


Table altered.


Table altered.


Table altered.


1 row created.


1 row created.


1 row created.


1 row created.


1 row created.


PRODUCT_ID PRODUCT_NAME
---------- --------------------
CATEGORY
------------------------------
         1 ThinkPad
Electronic

         2 Mouse
Electronic

         3 Office Chair
Office


PRODUCT_ID PRODUCT_NAME
---------- --------------------
CATEGORY
------------------------------
         4 Camera
Electronic

         5 Keyboard
Electronic
```

```
1 row created.


1 row created.


1 row created.


1 row created.


1 row created.


  STORE_ID STORE_NAME
---------- --------------------------------
ADDRESS
----------------------------------------
         1 Freeland Choice
20 Avenue, Burwood, VIC3125

         2 City Bunnings
36 Maple street, Melbourne, VIC3000

         3 Burwood Electronic
20 Avenue, Burwood


  STORE_ID STORE_NAME
---------- --------------------------------
ADDRESS
----------------------------------------
         4 Burwood Officework
606 Burwood Hwy, Vermont South VIC 3133

         5 Burwood Bunnings
606-634 Burwood Hwy, Vermont South VIC 3133
```

```
1 row created.


1 row created.


COMPANY_ID COMPANY_CONTACT
---------- ------------------
COMPANY_NAME
------------------------------
         1 111111
IBM

         2 222222
Luxo Narelle Mesh
```

```
1 row created.


1 row created.


1 row created.


CUSTOMER_ID CUSTOMER_NAME
----------- ------------------------------------------------
PAYMENT_TYPE                                       COMPANY_ID
-------------------------------------------------- ----------
          1 Kevin Li
Visa                                                        1

          2 David Andrews
Mastercard                                                  1

          3 Daniel Andrews
Mastercard                                                  2
```

```
1 row created.


1 row created.

1 row created.


1 row created.


1 row created.

1 row created.


1 row created.


ORDER_NUMBER   STORE_ID CUSTOMER_ID PRODUCT_ID     AMOUNT      PRICE TRANSACTI
------------ ---------- ----------- ---------- ---------- ---------- ---------
           1          1           1          1          1       1500 21-MAR-21
           2          1           1          2          2         30 21-MAR-21
           3          2           3          3          5         79 21-MAR-21
           4          3           1          4          1        100 22-MAR-21
           5          3           1          5          2         50 22-MAR-21
           6          4           2          1          2       1350 23-MAR-21
           7          5           3          3          4         80 24-MAR-21

7 rows selected.


Table altered.


Table altered.


Table altered.


Table altered.
```