데이터베이스시스템

(CSE4110-02)

Project 2. Normalization and Query Processing

서강대학교 컴퓨터공학과
20181614
김주연

목차

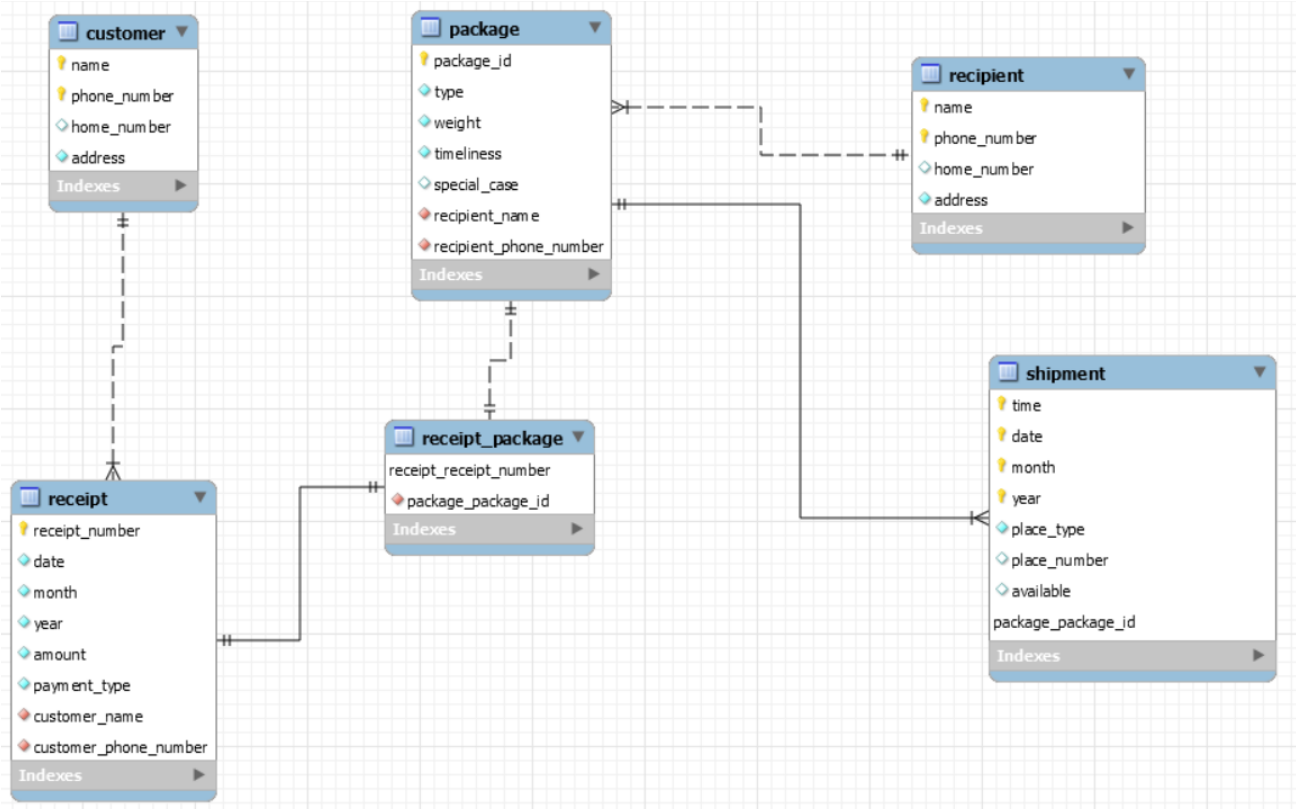
1. 프로그램 개요

1.1 프로젝트 목표

The goal of this project is to provide a realistic experience in the physical design, query processing implementation and maintenance of a small relational database you made in Project 1.

2. BCNF Decomposition

2.1 Logical Schema Diagram



2.2 Entity

- customer

customer
name(PK) phone_number(PK) home_number address

기존 customer entity에서는 주소를 composite attribute로 나타내었으나, 현재 상황에서는 주소를 매우 상세하게 나타낼 필요는 없을 것 같아 주소를 address의 simple attribute 하나로 나타내도록 하였다.

customer에는 name, phone_number -> home_number, address의 functional dependency가 존재한다.

- receipt

receipt
receipt_number(PK) date month year amount payment_type name(FK) phone_number(FK)

기존 receipt entity에서는 package_id를 FK로 가지고 있었으나 현재는 receipt-package 관계를 table로 만들었기 때문에 package_id를 포함하지 않는다.
recipient에는 receipt_number -> date, month, year, amount, payment_type, customer_name(FK), customer_phone_number(FK) 의 functional dependency가 존재한다.

- package

package
package_id(PK) type weight timeliness special_case name(FK) phone_number(FK)

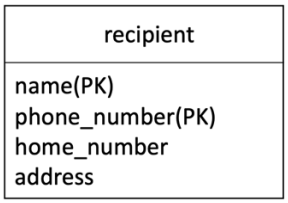
package에는 package_id -> type, weight, timeliness, special_case, recipient_name(FK), recipient_phone_number(FK) 의 functional dependency가 존재한다.

- shipment

shipment
package_id(PK, FK) time(PK) date(PK) month(PK) year(PK) place_type place_number available

shipment에는 package_package_id(FK), time, date, month, year -> place_type, place_number, available 의 functional dependency가 존재한다.

- recipient

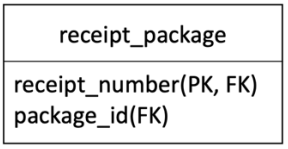


기존 recipient entity에서는 주소를 composite attribute로 나타내었으나, 현재 상황에서는 주소를 매우 상세하게 나타낼 필요는 없을 것 같아 주소를 address의 simple attribute 하나로 나타내도록 하였다.

recipient에는 name, phone_number -> home_number, address 의 functional dependency가 존재한다.

2.3 Relationship

- receipt_package

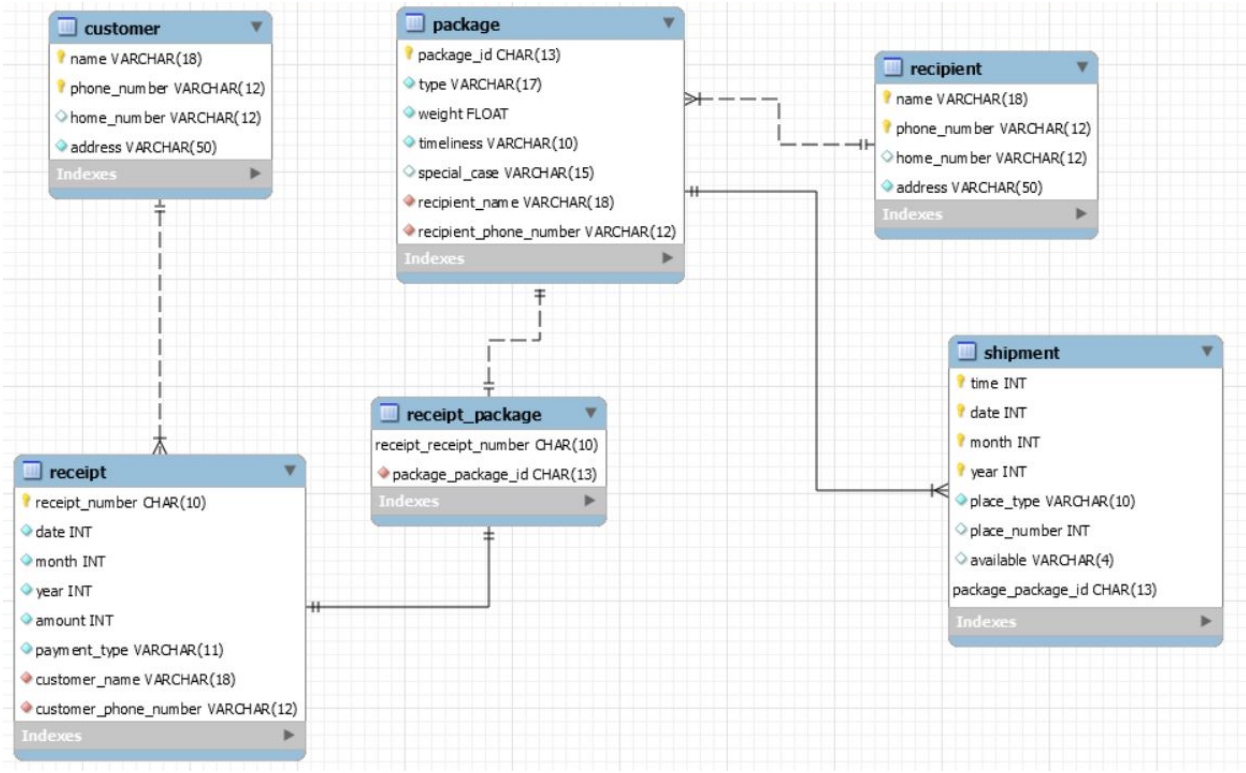


receipt와 package가 일대일 대응 관계이기 때문에 따로 table을 만들었다.

receipt_package에는 receipt_receipt_numbe(FK) -> packge_package_id(FK) 의 functional dependency가 존재한다.

이외의 relation들은 따로 table을 만들 필요가 없으므로 만들지 않았다.

3. Physical Schema Diagram



- receipt_number는 해당 택배회사에서 모두 동일하게 적용되어야 하는 것이므로 글자수 10자로 제한을 두었다. package_id 또한 해당 택배회사에서 모두 동일하게 적용되어야 하는 것이므로 글자수 13으로 제한을 두었다. 이를 제외한 문자열 입력을 받는 항목들은 최댓값만 정해 주어 융통성 있는 입력이 가능하도록 하였다.
- customer와 recipient의 home_number는 실제 배송에서 부가적인 항목이므로 null 값을 허용하였다.
- package의 special_case는 'Hazardous', 'International'등 특수한 경우를 위해 만든 항목이므로 null 값을 허용하였다.
  (추가적으로, package에서 type은 package의 type을 문자열로 저장하고 있으며 'Flat envelope', 'Small box', 'Large box' 등의 종류가 있다. weight는 package의 무게를 숫자로 저장한다. timeliness는 택배의 배송 타입을 문자열로 저장하고 있으며, 'Overnight', 'Second day' 등이 있다.)
- shipment의 place_number와 available은 해당 택배가 이미 수취자에게 도착했다면 나타낼 필요가 없는 항목이므로(place_number는 창고나 비행기, 트럭의 번호를 적는 항목이며 available은 해당 장소가 현재 제대로 작동하고 있는지 나타내는 용도이므로) null 값을 허용하였다.
  (추가적으로, shipment에서 place_type은 해당 시점에서의 택배의 위치를 문자열로 저장한다. place_type에 해당하는 항목에는 'Truck', 'Warehouse', 'Recipient' 등이 있다.)

4. Queries

4.1. Type 1 Assume truck 1721 is destroyed in a crash.

4.1.1. Type 1-1

Find all customers who had a package on the truck at the time of the crash.

```
select receipt.name, receipt.phone_number
from receipt, receipt_package, shipment
where receipt.receipt_number = receipt_package.receipt_number and
        receipt_package.package_id = shipment.package_id and
        shipment.place_type = "Truck" and shipment.place_number = "1721" and available = "No"
```

shipment는 배송 현황에 관한 정보를 저장하고 있다. shipment의 tuple 중 place_type='Truck', place_number='1721', available='No' 인 항목에 해당하는 택배 번호를 가리키는 접수 번호를 가지는 customer를 select하면 해당 트럭에 package가 들어 있는 customer를 찾을 수 있다. customer는 동명이인이 있을 수 있으므로 name과 phone_number를 함께 나타낸다.

4.1.2. Type 1-2

Find all recipients who had a package on that truck at the time of the crash.

```
select package.name, package.phone_number
from package, receipt_package, shipment
where package.package_id = receipt_package.package_id and
        receipt_package.package_id = shipment.package_id and
        shipment.place_type = "Truck" and shipment.place_number = "1721" and available = "No"
```

shipment의 tuple 중 place_type='Truck', place_number='1721', available='No' 인 항목에 해당하는 택배 번호를 가지는 recipient를 select하면 해당 트럭에 package가 들어 있는 recipient를 찾을 수 있다. recipient는 동명이인이 있을 수 있으므로 name과 phone_number를 함께 나타낸다.

4.1.3. Type 1-3

Find the last successful delivery by that truck prior to the crash.

```
select T.id
from (select shipment.package_id as id, shipment.date, shipment.month, shipment.year
        from package, shipment, receipt_package
        where package.package_id = receipt_package.package_id and
                receipt_package.package_id = shipment.package_id and
                shipment.place_type = "Truck" and shipment.place_number = "1721" and available = "Yes") as T
order by T.year desc, T.month desc, T.date desc
limit 1
```

shipment의 tuple 중 place_type='Truck', place_number='1721', available='Yes(트럭이 고장나지 않은 상태일때 이송된 택배를 뜻함)' 에 해당하는 택배들을 year,

month, date 순서로 ordering하여서 가장 최근의 항목 하나만 select하면 해당 질의의 답을 찾을 수 있다.

4.2. Type 2

Find the customer who has shipped the most packages in the past certain year.

```
select receipt.name, receipt.phone_number, count(receipt_number) as receipt_count
from receipt
where receipt.year = (stdin)
group by receipt.phone_number
order by receipt_count desc
limit 1
```

receipt entity는 접수 정보와 접수자(발송자) 정보를 저장하고 있다. 따라서 receipt의 tuple을 customer 정보에 의해 grouping하여 접수한 수를 count하고 그 수로 ordering하여서 가장 큰 항목 하나만 select하면 가장 많은 package를 보낸 customer를 찾을 수 있다. customer는 동명이인이 있을 수 있으므로 name과 phone_number를 함께 나타낸다.

4.3. Type 3

Find the customer who has spent the most money on shipping in the past certain year.

```
select receipt.name, receipt.phone_number, sum(receipt.amount) as receipt_money
from receipt
where receipt.year = (stdin)
group by receipt.phone_number
order by receipt_money desc
limit 1
```

receipt entity는 접수 정보와 접수자(발송자) 정보를 저장하고 있다. 따라서 receipt의 tuple을 customer 정보에 의해 grouping하고, 접수 금액의 sum을 구하여 그 수로 ordering하여서 가장 큰 항목 하나만 select하면 가장 많은 돈을 소비한 customer를 찾을 수 있다. customer는 동명이인이 있을 수 있으므로 name과 phone_number를 함께 나타낸다.

4.4. Type 4

Find those packages that were not delivered within the promised time.

```
select distinct T.package_id
from( select package.package_id, shipment.date as c_date, shipment.place_type
from package, shipment, receipt, receipt_package
where shipment.place_type = "Recipient" and shipment.package_id = package.package_id) as T, receipt_package, receipt, package
where T.package_id = receipt_package.package_id
    and receipt_package.receipt_number = receipt.receipt_number
    and package.timeliness = "Overnight" and (receipt.date+1<T.c_date)
union
select distinct T.package_id
from( select package.package_id, shipment.date as c_date, shipment.place_type
from package, shipment, receipt, receipt_package
where shipment.place_type = "Recipient" and shipment.package_id = package.package_id) as T, receipt_package, receipt, package
where T.package_id = receipt_package.package_id
    and receipt_package.receipt_number = receipt.receipt_number
    and package.timeliness = "Second day" and (receipt.date+2<T.c_date)
union
select distinct T.package_id
from( select package.package_id, shipment.date as c_date, shipment.place_type
from package, shipment, receipt, receipt_package
where shipment.place_type = "Recipient" and shipment.package_id = package.package_id) as T, receipt_package, receipt, package
where T.package_id = receipt_package.package_id
    and receipt_package.receipt_number = receipt.receipt_number
    and package.timeliness = "Third day" and (receipt.date+3<T.c_date)
```

shipment에서 place_type='Recipient'인 tuple을 찾으면 배송 완료 시점을 찾을 수 있다. 해당 shipment의 package_id와 대응되는 receipt_number를 가지는 receipt entity는 해당 택배가 접수된 날짜를 가지고 있으므로 이를 배송 시작 시점으로 생각할 수 있다. 배송 시작 시점과 완료 시점의 차이가 해당 package_id를 가지는 택배의 timeliness를 초과하는지를 비교하면 배송 약속일 이후에 도착한 택배를 찾을 수 있다. <5. Implementation>에서 사용한 sample data는 Timeliness가 Third day까지 포함하도록 하였으므로, Timeliness가 Overnight인 경우와 Second day인 경우, Third day인 경우의 배송 약속일 이후에 도착한 택배를 각각 찾아서 union하는 방식으로 질의의 답을 찾을 수 있다.

4.5. Type 5

Generate the bill for each customer for the past certain month.

Consider creating several types of bills.

- A simple bill: customer, address, and amount owed.

```
select customer.name, customer.phone_number, customer.address, sum(receipt.amount)
from customer, receipt
where customer.name = receipt.name and customer.phone_number = receipt.phone_number
        and customer.name = (stdin)
        and customer.phone_number = (stdin)
        and receipt.year = (stdin)
        and receipt.month = (stdin)
```

stdin으로 이름과 휴대폰 번호(동명이인이 있을 수 있으므로), 날짜를 받아서 customer에서 주소를 찾고, receipt에서 date(year, month) 가 입력과 일치하는 항목을 찾아 지불 금액인 amount의 sum을 구하면 그 달에 지불한 총 금액을 찾아

simple bill을 구성할 수 있다.

- A bill listing charges by type of service.

- An itemize billing listing each individual shipment and the charges for it.

```
select package.package_id, receipt.amount, package.type, receipt.payment_type, package.timeliness
from customer, receipt, package, receipt_package
where customer.name = receipt.name
        and customer.phone_number = receipt.phone_number
        and customer.name = (stdin)
        and customer.phone_number = (stdin)
        and receipt.year = (stdin)
        and receipt.month = (stdin)
        and receipt.receipt_number = receipt_package.receipt_number
        and receipt_package.package_id = package.package_id
```

stdin으로 받은 이름과 휴대폰 번호, 날짜를 통해서 접수일이 입력받은 year, month와 일치하는 receipt_number와 amount를 찾을 수 있고, receipt_package에서 receipt_number에 대응되는 package_id를 구할 수 있다. 이후 package에서 package_id로 택배 정보를 찾아서 출력하면 itemize bill을 구성할 수 있다.

5. Implementation

- **sample data (각 테이블은 15개 이상의 tuple로 구성하였음)**

name	phone_number	home_number	address
Byeon	01000000011	null	A
Cheon	01000000015	null	F
Choi	01000000004	null	B
Jeon	01000000006	null	A
Kang	01000000005	null	C
Kim	01000000001	null	A
Kim	01000000009	null	E
Kwon	01000000013	null	B
Lee	01000000003	null	B
Lee	01000000008	null	D
Lim	01000000014	null	E
Na	01000000012	null	G
Noh	01000000007	null	G
Park	01000000002	null	A
Park	01000000010	null	F
NULL	NULL	NULL	NULL

(customer)

package_id	type	weight	timeliness	special_case	name	phone_number
DB12345678900	Flat envelopment	1.2	Overnight	null	Alice	01011111101
DB12345678901	Small box	1.6	Overnight	Hazardous	Bella	01011111115
DB12345678902	Flat envelopment	1.2	Second day	null	Alice	01011111101
DB12345678903	Flat envelopment	1.3	Overnight	Hazardous	Chalie	01011111104
DB12345678904	Small box	2.1	Second day	null	Chloe	01011111114
DB12345678905	Large box	3.4	Third day	International	Chris	01011111111
DB12345678906	Large box	4.7	Overnight	Hazardous	Dean	01011111113
DB12345678907	Small vox	2.5	Second day	null	James	01011111106
DB12345678908	Flat envelopment	1.7	Second day	Hazardous	Jane	01011111102
DB12345678909	Flat envelopment	1.5	Overnight	null	Jenny	01011111110
DB12345678910	Flat envelopment	1.2	Overnight	null	Jessica	01011111108
DB12345678911	Small box	2.1	Second day	null	Mary	01011111112
DB12345678912	Large box	3.5	Third day	International	Mina	01011111109
DB12345678913	Large box	4.4	Overnight	null	Troye	01011111105
DB12345678914	Small box	2.5	Overnight	null	Alex	01011111107
DB12345678915	Flat envelopment	1.5	Second day	Hazardous	Ben	01011111103
DB12345678916	Small box	1.8	Overnight	null	Jane	01011111102
NULL	NULL	NULL	NULL	NULL	NULL	NULL

(package)

receipt_number	package_id
1234567800	DB12345678900
1234567801	DB12345678901
1234567802	DB12345678902
1234567803	DB12345678903
1234567804	DB12345678904
1234567805	DB12345678905
1234567806	DB12345678906
1234567807	DB12345678907
1234567808	DB12345678908
1234567809	DB12345678909
1234567810	DB12345678910
1234567811	DB12345678911
1234567812	DB12345678912
1234567813	DB12345678913
1234567814	DB12345678914
1234567815	DB12345678915
1234567816	DB12345678916
NULL	NULL

(receipt_Package)

receipt_number	date	month	year	amount	payment_type	name	phone_number
1234567800	22	5	2020	2300	Monthly	Kang	01000000005
1234567801	21	5	2020	3400	Credit	Cheon	01000000015
1234567802	15	5	2020	2300	Prepaid	Choi	01000000004
1234567803	28	4	2020	2300	Monthly	Jeon	01000000006
1234567804	12	4	2020	3500	Monthly	Kang	01000000005
1234567805	15	3	2020	4700	Credit	Kim	01000000001
1234567806	5	3	2020	4500	Prepaid	Kim	01000000001
1234567807	2	3	2020	3300	Prepaid	Kwon	01000000013
1234567808	17	1	2020	2100	Prepaid	Kang	01000000005
1234567809	22	12	2019	2000	Credit	Lee	01000000003
1234567810	24	10	2019	2400	Monthly	Lim	01000000014
1234567811	18	9	2019	3400	Credit	Na	01000000012
1234567812	12	8	2019	8400	Monthly	Noh	01000000007
1234567813	4	8	2019	4800	Monthly	Park	01000000002
1234567814	19	9	2019	3200	Prepaid	Park	01000000002
1234567815	16	7	2019	2600	Monthly	Lee	01000000008
1234567816	2	7	2018	3300	Prepaid	Byeon	01000000011
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(receipt)

name	phone_number	home_number	address
Alex	01011111107	null	E
Alice	01011111101	null	A
Bella	01011111115	null	B
Ben	01011111103	null	A
Chalie	01011111104	null	F
Chloe	01011111114	null	A
Chris	01011111111	null	F
Dean	01011111113	null	A
James	01011111106	null	D
Jane	01011111102	null	D
Jenny	01011111110	null	E
Jessica	01011111108	null	A
Mary	01011111112	null	D
Mina	01011111109	null	F
Troye	01011111105	null	D
NULL	NULL	NULL	NULL

(recipient)

time	date	month	year	place_type	place_number	available	package_id
11	28	4	2020	Warehouse	2432	Yes	DB12345678903
12	22	5	2020	Warehouse	2332	Yes	DB12345678900
12	22	5	2020	Warehouse	2332	Yes	DB12345678901
13	15	4	2020	Recipient	0	null	DB12345678904
13	16	5	2020	Truck	1721	Yes	DB12345678902
13	23	12	2019	Recipient	0	null	DB12345678909
14	18	1	2020	Recipient	0	null	DB12345678908
14	18	5	2020	Recipient	0	null	DB12345678902
15	7	3	2020	Recipient	0	null	DB12345678906
15	22	5	2020	Truck	1721	No	DB12345678900
15	22	5	2020	Truck	1721	No	DB12345678901
15	22	12	2019	Warehouse	2432	Yes	DB12345678909
15	29	4	2020	Truck	2661	Yes	DB12345678903
16	5	3	2020	Truck	5521	Yes	DB12345678906
16	6	3	2020	Truck	1721	Yes	DB12345678906
16	15	5	2020	Warehouse	2332	Yes	DB12345678902
16	16	3	2020	Recipient	0	null	DB12345678905
17	29	4	2020	Recipient	0	null	DB12345678903
18	3	3	2020	Recipient	0	null	DB12345678907
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(shipment)

- **case**

```
****************************************
******** 20181614 김주연 DBProject2 ********
****************************************


------- SELECT QUERY TYPES -------

        1. TYPE I
        2. TYPE II
        3. TYPE III
        4. TYPE IV
        5. TYPE V
        0. QUIT

Which type of query?
```

Main menu

5.1. Type 1

```
-------- SELECT QUERY TYPES --------

        1. TYPE I
        2. TYPE II
        3. TYPE III
        4. TYPE IV
        5. TYPE V
        0. QUIT

Which type of query? 1


---- TYPE I ----

Input the number of truck : 1366
Truck 1366 is not destroyed.

Input the number of truck : 1721

------- Subtypes in TYPE I -------

        1. TYPE I-1
        2. TYPE I-2
        3. TYPE I-3
        0. QUIT
Which type of query?
```

Truck number로 1721이 들어오기 전까지 계속 입력받는다.

5.1.1. Type 1-1

```
------- Subtypes in TYPE I -------

        1. TYPE I-1
        2. TYPE I-2
        3. TYPE I-3
        0. QUIT
Which type of query? 1

---- TYPE I-1 ----

** Find all customers who had a package on the truck at the time of the crash. **
Customer name : Kang 01000000005  Cheon 01000000015
```

time	date	month	year	place_type	place_number	available	package_id
11	28	4	2020	Warehouse	2432	Yes	DB12345678903
12	22	5	2020	Warehouse	2332	Yes	DB12345678900
12	22	5	2020	Warehouse	2332	Yes	DB12345678901
13	15	4	2020	Recipient	0	null	DB12345678904
13	16	5	2020	Truck	1721	Yes	DB12345678902
13	23	12	2019	Recipient	0	null	DB12345678909
14	18	1	2020	Recipient	0	null	DB12345678908
14	18	5	2020	Recipient	0	null	DB12345678902
15	7	3	2020	Recipient	0	null	DB12345678906
15	22	5	2020	Truck	1721	No	DB12345678900
15	22	5	2020	Truck	1721	No	DB12345678901
15	22	12	2019	Warehouse	2432	Yes	DB12345678909
15	29	4	2020	Truck	2661	Yes	DB12345678903
16	5	3	2020	Truck	5521	Yes	DB12345678906
16	6	3	2020	Truck	1721	Yes	DB12345678906
16	15	5	2020	Warehouse	2332	Yes	DB12345678902
16	16	3	2020	Recipient	0	null	DB12345678905
17	29	4	2020	Recipient	0	null	DB12345678903
18	3	3	2020	Recipient	0	null	DB12345678907
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(shipment)

receipt_number	package_id
1234567800	DB12345678900
1234567801	DB12345678901
1234567802	DB12345678902
1234567803	DB12345678903
1234567804	DB12345678904
1234567805	DB12345678905
1234567806	DB12345678906
1234567807	DB12345678907
1234567808	DB12345678908
1234567809	DB12345678909
1234567810	DB12345678910
1234567811	DB12345678911
1234567812	DB12345678912
1234567813	DB12345678913
1234567814	DB12345678914
1234567815	DB12345678915
1234567816	DB12345678916
NULL	NULL

(receipt_package)

receipt_number	date	month	year	amount	payment_type	name	phone_number
1234567800	22	5	2020	2300	Monthly	Kang	01000000005
1234567801	21	5	2020	3400	Credit	Cheon	01000000015
1234567802	15	5	2020	2300	Prepaid	Choi	01000000004
1234567803	28	4	2020	2300	Monthly	Jeon	01000000006
1234567804	12	4	2020	3500	Monthly	Kang	01000000005
1234567805	15	3	2020	4700	Credit	Kim	01000000001
1234567806	5	3	2020	4500	Prepaid	Kim	01000000001
1234567807	2	3	2020	3300	Prepaid	Kwon	01000000013
1234567808	17	1	2020	2100	Prepaid	Kang	01000000005
1234567809	22	12	2019	2000	Credit	Lee	01000000003
1234567810	24	10	2019	2400	Monthly	Lim	01000000014
1234567811	18	9	2019	3400	Credit	Na	01000000012
1234567812	12	8	2019	8400	Monthly	Noh	01000000007
1234567813	4	8	2019	4800	Monthly	Park	01000000002
1234567814	19	9	2019	3200	Prepaid	Park	01000000002
1234567815	16	7	2019	2600	Monthly	Lee	01000000008
1234567816	2	7	2018	3300	Prepaid	Byeon	01000000011
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(customer)

5.1.2. Type 1-2

```
-------- Subtypes in TYPE I --------

        1. TYPE I-1
        2. TYPE I-2
        3. TYPE I-3
        0. QUIT
Which type of query? 2

---- TYPE I-2 ----

** Find all recipients who had a package on that truck at the time of the crash. **
Recipient name : Alice 01011111101  Bella 01011111115
```

time	date	month	year	place_type	place_number	available	package_id
11	28	4	2020	Warehouse	2432	Yes	DB12345678903
12	22	5	2020	Warehouse	2332	Yes	DB12345678900
12	22	5	2020	Warehouse	2332	Yes	DB12345678901
13	15	4	2020	Recipient	0	null	DB12345678904
13	16	5	2020	Truck	1721	Yes	DB12345678902
13	23	12	2019	Recipient	0	null	DB12345678909
14	18	1	2020	Recipient	0	null	DB12345678908
14	18	5	2020	Recipient	0	null	DB12345678902
15	7	3	2020	Recipient	0	null	DB12345678906
15	22	5	2020	Truck	1721	No	DB12345678900
15	22	5	2020	Truck	1721	No	DB12345678901
15	22	12	2019	Warehouse	2432	Yes	DB12345678909
15	29	4	2020	Truck	2661	Yes	DB12345678903
16	5	3	2020	Truck	5521	Yes	DB12345678906
16	6	3	2020	Truck	1721	Yes	DB12345678906
16	15	5	2020	Warehouse	2332	Yes	DB12345678902
16	16	3	2020	Recipient	0	null	DB12345678905
17	29	4	2020	Recipient	0	null	DB12345678903
18	3	3	2020	Recipient	0	null	DB12345678907
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(shipment)

package_id	type	weight	timeliness	special_case	name	phone_number
DB12345678900	Flat envelopment	1.2	Overnight	null	Alice	01011111101
DB12345678901	Small box	1.6	Overnight	Hazardous	Bella	01011111115
DB12345678902	Flat envelopment	1.2	Second day	null	Alice	01011111101
DB12345678903	Flat envelopment	1.3	Overnight	Hazardous	Chalie	01011111104
DB12345678904	Small box	2.1	Second day	null	Chloe	01011111114
DB12345678905	Large box	3.4	Third day	International	Chris	01011111111
DB12345678906	Large box	4.7	Overnight	Hazardous	Dean	01011111113
DB12345678907	Small vox	2.5	Second day	null	James	01011111106
DB12345678908	Flat envelopment	1.7	Second day	Hazardous	Jane	01011111102
DB12345678909	Flat envelopment	1.5	Overnight	null	Jenny	01011111110
DB12345678910	Flat envelopment	1.2	Overnight	null	Jessica	01011111108
DB12345678911	Small box	2.1	Second day	null	Mary	01011111112
DB12345678912	Large box	3.5	Third day	International	Mina	01011111109
DB12345678913	Large box	4.4	Overnight	null	Troye	01011111105
DB12345678914	Small box	2.5	Overnight	null	Alex	01011111107
DB12345678915	Flat envelopment	1.5	Second day	Hazardous	Ben	01011111103
DB12345678916	Small box	1.8	Overnight	null	Jane	01011111102
NULL	NULL	NULL	NULL	NULL	NULL	NULL

(package)

5.1.3. Type 1-3

```
------- Subtypes in TYPE I -------

        1. TYPE I-1
        2. TYPE I-2
        3. TYPE I-3
        0. QUIT
Which type of query? 3

---- TYPE I-3 ----

** Find the last successful delivery by that truck prior to the crash. **
Delivery ID : DB12345678902
```

time	date	month	year	place_type	place_number	available	package_id
11	28	4	2020	Warehouse	2432	Yes	DB12345678903
12	22	5	2020	Warehouse	2332	Yes	DB12345678900
12	22	5	2020	Warehouse	2332	Yes	DB12345678901
13	15	4	2020	Recipient	0	null	DB12345678904
13	16	5	2020	Truck	1721	Yes	DB12345678902
13	23	12	2019	Recipient	0	null	DB12345678909
14	18	1	2020	Recipient	0	null	DB12345678908
14	18	5	2020	Recipient	0	null	DB12345678902
15	7	3	2020	Recipient	0	null	DB12345678906
15	22	5	2020	Truck	1721	No	DB12345678900
15	22	5	2020	Truck	1721	No	DB12345678901
15	22	12	2019	Warehouse	2432	Yes	DB12345678909
15	29	4	2020	Truck	2661	Yes	DB12345678903
16	5	3	2020	Truck	5521	Yes	DB12345678906
16	6	3	2020	Truck	1721	Yes	DB12345678906
16	15	5	2020	Warehouse	2332	Yes	DB12345678902
16	16	3	2020	Recipient	0	null	DB12345678905
17	29	4	2020	Recipient	0	null	DB12345678903
18	3	3	2020	Recipient	0	null	DB12345678907
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(shipment)

5.2. Type 2

```
------- SELECT QUERY TYPES -------

        1. TYPE I
        2. TYPE II
        3. TYPE III
        4. TYPE IV
        5. TYPE V
        0. QUIT

Which type of query? 2


---- TYPE II ----

** Find the customer who has shipped the most packages in certain year **
 Which Year? : 2020
Customer name : Kang 01000000005

** Find the customer who has shipped the most packages in certain year **
 Which Year? : 2019
Customer name : Park 01000000002

** Find the customer who has shipped the most packages in certain year **
 Which Year? : 0
```

receipt_number	date	month	year	amount	payment_type	name	phone_number
1234567800	22	5	2020	2300	Monthly	Kang	01000000005
1234567801	21	5	2020	3400	Credit	Cheon	01000000015
1234567802	15	5	2020	2300	Prepaid	Choi	01000000004
1234567803	28	4	2020	2300	Monthly	Jeon	01000000006
1234567804	12	4	2020	3500	Monthly	Kang	01000000005
1234567805	15	3	2020	4700	Credit	Kim	01000000001
1234567806	5	3	2020	4500	Prepaid	Kim	01000000001
1234567807	2	3	2020	3300	Prepaid	Kwon	01000000013
1234567808	17	1	2020	2100	Prepaid	Kang	01000000005
1234567809	22	12	2019	2000	Credit	Lee	01000000003
1234567810	24	10	2019	2400	Monthly	Lim	01000000014
1234567811	18	9	2019	3400	Credit	Na	01000000012
1234567812	12	8	2019	8400	Monthly	Noh	01000000007
1234567813	4	8	2019	4800	Monthly	Park	01000000002
1234567814	19	9	2019	3200	Prepaid	Park	01000000002
1234567815	16	7	2019	2600	Monthly	Lee	01000000008
1234567816	2	7	2018	3300	Prepaid	Byeon	01000000011
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(receipt)

5.3. Type 3

```
-------- SELECT QUERY TYPES --------

        1. TYPE I
        2. TYPE II
        3. TYPE III
        4. TYPE IV
        5. TYPE V
        0. QUIT

Which type of query? 3


---- TYPE III ----

** Find the customer who has spent the most money on shipping in the past certain year **
 Which Year? : 2020
Customer name : Kim 01000000001

** Find the customer who has spent the most money on shipping in the past certain year **
 Which Year? : 2019
Customer name : Noh 01000000007

** Find the customer who has spent the most money on shipping in the past certain year **
 Which Year? : 2018
Customer name : Byeon 01000000011

** Find the customer who has spent the most money on shipping in the past certain year **
 Which Year? : 0
```

receipt_number	date	month	year	amount	payment_type	name	phone_number
1234567800	22	5	2020	2300	Monthly	Kang	01000000005
1234567801	21	5	2020	3400	Credit	Cheon	01000000015
1234567802	15	5	2020	2300	Prepaid	Choi	01000000004
1234567803	28	4	2020	2300	Monthly	Jeon	01000000006
1234567804	12	4	2020	3500	Monthly	Kang	01000000005
1234567805	15	3	2020	4700	Credit	Kim	01000000001
1234567806	5	3	2020	4500	Prepaid	Kim	01000000001
1234567807	2	3	2020	3300	Prepaid	Kwon	01000000013
1234567808	17	1	2020	2100	Prepaid	Kang	01000000005
1234567809	22	12	2019	2000	Credit	Lee	01000000003
1234567810	24	10	2019	2400	Monthly	Lim	01000000014
1234567811	18	9	2019	3400	Credit	Na	01000000012
1234567812	12	8	2019	8400	Monthly	Noh	01000000007
1234567813	4	8	2019	4800	Monthly	Park	01000000002
1234567814	19	9	2019	3200	Prepaid	Park	01000000002
1234567815	16	7	2019	2600	Monthly	Lee	01000000008
1234567816	2	7	2018	3300	Prepaid	Byeon	01000000011
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(receipt)

5.4. Type 4

```
-------- SELECT QUERY TYPES --------

        1. TYPE I
        2. TYPE II
        3. TYPE III
        4. TYPE IV
        5. TYPE V
        0. QUIT

Which type of query? 4


----- TYPE IV -----

** Find those packages that were not delivered within the promised time **
DB12345678904  DB12345678902  DB12345678906
```

time	date	month	year	place_type	place_number	available	package_id
11	28	4	2020	Warehouse	2432	Yes	DB12345678903
12	22	5	2020	Warehouse	2332	Yes	DB12345678900
12	22	5	2020	Warehouse	2332	Yes	DB12345678901
13	15	4	2020	Recipient	0	null	DB12345678904
13	16	5	2020	Truck	1721	Yes	DB12345678902
13	23	12	2019	Recipient	0	null	DB12345678909
14	18	1	2020	Recipient	0	null	DB12345678908
14	18	5	2020	Recipient	0	null	DB12345678902
15	7	3	2020	Recipient	0	null	DB12345678906
15	22	5	2020	Truck	1721	No	DB12345678900
15	22	5	2020	Truck	1721	No	DB12345678901
15	22	12	2019	Warehouse	2432	Yes	DB12345678909
15	29	4	2020	Truck	2661	Yes	DB12345678903
16	5	3	2020	Truck	5521	Yes	DB12345678906
16	6	3	2020	Truck	1721	Yes	DB12345678906
16	15	5	2020	Warehouse	2332	Yes	DB12345678902
16	16	3	2020	Recipient	0	null	DB12345678905
17	29	4	2020	Recipient	0	null	DB12345678903
18	3	3	2020	Recipient	0	null	DB12345678907
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(shipment)

receipt_number	package_id
1234567800	DB12345678900
1234567801	DB12345678901
1234567802	DB12345678902
1234567803	DB12345678903
1234567804	DB12345678904
1234567805	DB12345678905
1234567806	DB12345678906
1234567807	DB12345678907
1234567808	DB12345678908
1234567809	DB12345678909
1234567810	DB12345678910
1234567811	DB12345678911
1234567812	DB12345678912
1234567813	DB12345678913
1234567814	DB12345678914
1234567815	DB12345678915
1234567816	DB12345678916
NULL	NULL

(receipt_package)

receipt_number	date	month	year	amount	payment_type	name	phone_number
1234567800	22	5	2020	2300	Monthly	Kang	01000000005
1234567801	21	5	2020	3400	Credit	Cheon	01000000015
1234567802	15	5	2020	2300	Prepaid	Choi	01000000004
1234567803	28	4	2020	2300	Monthly	Jeon	01000000006
1234567804	12	4	2020	3500	Monthly	Kang	01000000005
1234567805	15	3	2020	4700	Credit	Kim	01000000001
1234567806	5	3	2020	4500	Prepaid	Kim	01000000001
1234567807	2	3	2020	3300	Prepaid	Kwon	01000000013
1234567808	17	1	2020	2100	Prepaid	Kang	01000000005
1234567809	22	12	2019	2000	Credit	Lee	01000000003
1234567810	24	10	2019	2400	Monthly	Lim	01000000014
1234567811	18	9	2019	3400	Credit	Na	01000000012
1234567812	12	8	2019	8400	Monthly	Noh	01000000007
1234567813	4	8	2019	4800	Monthly	Park	01000000002
1234567814	19	9	2019	3200	Prepaid	Park	01000000002
1234567815	16	7	2019	2600	Monthly	Lee	01000000008
1234567816	2	7	2018	3300	Prepaid	Byeon	01000000011
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(receipt)

package_id	type	weight	timeliness	special_case	name	phone_number
DB12345678900	Flat envelopment	1.2	Overnight	null	Alice	01011111101
DB12345678901	Small box	1.6	Overnight	Hazardous	Bella	01011111115
DB12345678902	Flat envelopment	1.2	Second day	null	Alice	01011111101
DB12345678903	Flat envelopment	1.3	Overnight	Hazardous	Chalie	01011111104
DB12345678904	Small box	2.1	Second day	null	Chloe	01011111114
DB12345678905	Large box	3.4	Third day	International	Chris	01011111111
DB12345678906	Large box	4.7	Overnight	Hazardous	Dean	01011111113
DB12345678907	Small vox	2.5	Second day	null	James	01011111106
DB12345678908	Flat envelopment	1.7	Second day	Hazardous	Jane	01011111102
DB12345678909	Flat envelopment	1.5	Overnight	null	Jenny	01011111110
DB12345678910	Flat envelopment	1.2	Overnight	null	Jessica	01011111108
DB12345678911	Small box	2.1	Second day	null	Mary	01011111112
DB12345678912	Large box	3.5	Third day	International	Mina	01011111109
DB12345678913	Large box	4.4	Overnight	null	Troye	01011111105
DB12345678914	Small box	2.5	Overnight	null	Alex	01011111107
DB12345678915	Flat envelopment	1.5	Second day	Hazardous	Ben	01011111103
DB12345678916	Small box	1.8	Overnight	null	Jane	01011111102
NULL	NULL	NULL	NULL	NULL	NULL	NULL

5.5. Type 5

```
------- SELECT QUERY TYPES -------

        1. TYPE I
        2. TYPE II
        3. TYPE III
        4. TYPE IV
        5. TYPE V
        0. QUIT

Which type of query? 5


---- TYPE V ----

** Generate the bill for each customer for the past certain month **
Customer name : Kim
Customer number : 01000000001
Which month(YYYY-MM)? : 2020-03
Generating Bill...

Generation Completed
```

bill_202003_Kim_01000000001.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
Customer        Phone_number    Address  Amount
Kim     01000000001     A       9200


-----------------------------
Itemized Billing List
| Package Number | Amount | Serivce Type | Payment Type | Timeliness of Delivery|
DB12345678905   4700    Large box       Credit   Third day
DB12345678906   4500    Large box       Prepaid  Overnight
```

receipt_number	date	month	year	amount	payment_type	name	phone_number
1234567800	22	5	2020	2300	Monthly	Kang	01000000005
1234567801	21	5	2020	3400	Credit	Cheon	01000000015
1234567802	15	5	2020	2300	Prepaid	Choi	01000000004
1234567803	28	4	2020	2300	Monthly	Jeon	01000000006
1234567804	12	4	2020	3500	Monthly	Kang	01000000005
1234567805	15	3	2020	4700	Credit	Kim	01000000001
1234567806	5	3	2020	4500	Prepaid	Kim	01000000001
1234567807	2	3	2020	3300	Prepaid	Kwon	01000000013
1234567808	17	1	2020	2100	Prepaid	Kang	01000000005
1234567809	22	12	2019	2000	Credit	Lee	01000000003
1234567810	24	10	2019	2400	Monthly	Lim	01000000014
1234567811	18	9	2019	3400	Credit	Na	01000000012
1234567812	12	8	2019	8400	Monthly	Noh	01000000007
1234567813	4	8	2019	4800	Monthly	Park	01000000002
1234567814	19	9	2019	3200	Prepaid	Park	01000000002
1234567815	16	7	2019	2600	Monthly	Lee	01000000008
1234567816	2	7	2018	3300	Prepaid	Byeon	01000000011
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

(receipt)

receipt_number	package_id
1234567800	DB12345678900
1234567801	DB12345678901
1234567802	DB12345678902
1234567803	DB12345678903
1234567804	DB12345678904
1234567805	DB12345678905
1234567806	DB12345678906
1234567807	DB12345678907
1234567808	DB12345678908
1234567809	DB12345678909
1234567810	DB12345678910
1234567811	DB12345678911
1234567812	DB12345678912
1234567813	DB12345678913
1234567814	DB12345678914
1234567815	DB12345678915
1234567816	DB12345678916
NULL	NULL

(receipt_package)

package_id	type	weight	timeliness	special_case	name	phone_number
DB12345678900	Flat envelopment	1.2	Overnight	null	Alice	01011111101
DB12345678901	Small box	1.6	Overnight	Hazardous	Bella	01011111115
DB12345678902	Flat envelopment	1.2	Second day	null	Alice	01011111101
DB12345678903	Flat envelopment	1.3	Overnight	Hazardous	Chalie	01011111104
DB12345678904	Small box	2.1	Second day	null	Chloe	01011111114
DB12345678905	Large box	3.4	Third day	International	Chris	01011111111
DB12345678906	Large box	4.7	Overnight	Hazardous	Dean	01011111113
DB12345678907	Small vox	2.5	Second day	null	James	01011111106
DB12345678908	Flat envelopment	1.7	Second day	Hazardous	Jane	01011111102
DB12345678909	Flat envelopment	1.5	Overnight	null	Jenny	01011111110
DB12345678910	Flat envelopment	1.2	Overnight	null	Jessica	01011111108
DB12345678911	Small box	2.1	Second day	null	Mary	01011111112
DB12345678912	Large box	3.5	Third day	International	Mina	01011111109
DB12345678913	Large box	4.4	Overnight	null	Troye	01011111105
DB12345678914	Small box	2.5	Overnight	null	Alex	01011111107
DB12345678915	Flat envelopment	1.5	Second day	Hazardous	Ben	01011111103
DB12345678916	Small box	1.8	Overnight	null	Jane	01011111102
NULL	NULL	NULL	NULL	NULL	NULL	NULL

(package)

- **Code (상세한 내용은 20181614.c 파일에 주석으로 설명하였음)**

```c
    printf("*****************************************₩n");
    printf("******** 20181614 김주연 DBProject2 ********₩n");
    printf("*****************************************₩n₩n");

    const char* query;
    int state = 0;

    int type;
    int type_one = 1, truck_num = 1;
    char year[5], month[3], name[10];
    char *str, *tmp;
    FILE* fp = fopen("db_setup.txt", "r");  // db_setup.txt에는 table create, update 코드가 있음
    while (!feof(fp)) {                     // 파일에서 한 줄씩 읽어와서 데이터베이스 초기화 수행
        str = (char*)malloc(500 * sizeof(char));
        fgets(str, 500, fp);

        state = mysql_query(connection, str);
        if (state != 0) continue;
        sql_result = mysql_store_result(connection);
        mysql_free_result(sql_result);
        free(str);
    }
    fclose(fp);
    while (1) {                             // 입력으로 0을 받기 전까지 while loop 수행
        printf("₩n------- SELECT QUERY TYPES -------₩n₩n");
        printf("₩t1. TYPE I₩n");
        printf("₩t2. TYPE II₩n");
        printf("₩t3. TYPE III₩n");
        printf("₩t4. TYPE IV₩n");
        printf("₩t5. TYPE V₩n");
        printf("₩t0. QUIT₩n₩n");
        printf("Which type of query? ");
        scanf("%d", &type);
        printf("₩n₩n");

        if(type==1){
            printf("---- TYPE I ----₩n₩n");     //  Assume truck 1721 is destroyed in a crash.
            while(truck_num!=0){
                printf("Input the number of truck : ");
                scanf("%d", &truck_num);
                if (truck_num == 0) break;      // truck number 0이면 back, 1721이 아니면 재입력 받음
                else if (truck_num != 1721) printf("Truck %d is not destroyed.₩n₩n", truck_num);
                else {                          // truck number 1721인 경우 type 1-1, 1-2, 1-3 수행
                    while (1) {
                        printf("₩n------- Subtypes in TYPE I -------₩n₩n");
                        printf("₩t1. TYPE I-1₩n");
                        printf("₩t2. TYPE I-2₩n");
                        printf("₩t3. TYPE I-3₩n");
                        printf("₩t0. QUIT₩n");
                        printf("Which type of query? ");
                        scanf("%d", &type_one);
                        if (type_one == 0) break;
                        if (type_one == 1) {
                            printf("₩n---- TYPE I-1 ----₩n₩n");
                            printf("** Find all customers who had a package on the truck at the time of the crash. **₩n");
                            fp = fopen("type1-1.txt", "r");     // txt 파일에 저장된 query 읽어옴
                            while (!feof(fp)) {
                                str = (char*)malloc(500 * sizeof(char));
                                fgets(str, 500, fp);
                                printf("Customer name : ");
                                state = mysql_query(connection, str);
                                if (state != 0) continue;
                                sql_result = mysql_store_result(connection);
                                while ((sql_row = mysql_fetch_row(sql_result)) != NULL)
                                {
                                    printf("%s %s  ", sql_row[0], sql_row[1]);  // customer는 name, phone_number로 구분되므로 둘 다 표시
                                }
                                printf("₩n₩n");
                                mysql_free_result(sql_result);
                                free(str);
                            }
                            fclose(fp);
                        }
                        else if (type_one == 2) {
                            printf("₩n---- TYPE I-2 ----₩n₩n");
                            printf("** Find all recipients who had a package on that truck at the time of the crash. **₩n");
                            fp = fopen("type1-2.txt", "r");     // txt 파일에 저장된 query 읽어옴
                            while (!feof(fp)) {
                                str = (char*)malloc(500 * sizeof(char));
                                fgets(str, 500, fp);
                                printf("Recipient name : ");
                                state = mysql_query(connection, str);
                                if (state != 0) continue;
                                sql_result = mysql_store_result(connection);
                                while ((sql_row = mysql_fetch_row(sql_result)) != NULL)
                                {
                                    printf("%s %s  ", sql_row[0], sql_row[1]);  // recipient는 name, phone_number로 구분되므로 둘 다 표시
                                }
                                printf("₩n₩n");
                                mysql_free_result(sql_result);
                                free(str);
                            }
                            fclose(fp);
                        }
```

```
                    else if (type_one == 3) {
                        printf("₩n---- TYPE I-3 ----₩n₩n");
                        printf("** Find the last successful delivery by that truck prior to the crash. **₩n");
                        fp = fopen("type1-3.txt", "r");     // txt 파일에 저장된 query 읽어옴
                        while (!feof(fp)) {
                            str = (char*)malloc(500 * sizeof(char));
                            fgets(str, 500, fp);
                            printf("Delivery ID : ");
                            state = mysql_query(connection, str);
                            if (state != 0) continue;
                            sql_result = mysql_store_result(connection);
                            while ((sql_row = mysql_fetch_row(sql_result)) != NULL)
                            {
                                printf("%s  ", sql_row[0]); // package는 package_id로 구분되므로 package_id만 표시
                            }
                            printf("₩n₩n");
                            mysql_free_result(sql_result);
                            free(str);
                        }
                        fclose(fp);
                    }
                    else continue;
                }
                if (type_one == 0) break;
            }
        }
    }
    else if (type==2) {
        printf("---- TYPE II ----₩n₩n");
        while (1) {
            printf("** Find the customer who has shipped the most packages in certain year **₩n");
            printf(" Which Year? : ");
            scanf("%s", year);                      // stdin으로 연도 입력받음
            if (!strcmp(year,"0")) break;
            else {
                fp = fopen("type2.txt", "r");       // txt 파일에 저장된 query 읽어옴
                str = (char*)malloc(500 * sizeof(char));
                tmp = (char*)malloc(100 * sizeof(char));
                fgets(str, 500, fp);                // 입력받은 연도를 query에 넣어주는 과정
                strcat(str, year);
                fgets(tmp, 500, fp);
                strcat(str, tmp);
                printf("Customer name : ");
                state = mysql_query(connection, str);
                if (state != 0) continue;
                sql_result = mysql_store_result(connection);
                while ((sql_row = mysql_fetch_row(sql_result)) != NULL)
                {
                    printf("%s %s  ", sql_row[0], sql_row[1]);  // customer는 name, phone_number로 구분되므로 둘 다 표시
                }
                printf("₩n₩n");
                mysql_free_result(sql_result);
                free(str);
                fclose(fp);
            }
        }
    }
    else if (type == 3) {
        printf("---- TYPE III ----₩n₩n");
        while (1) {
            printf("** Find the customer who has spent the most money on shipping in the past certain year **₩n");
            printf(" Which Year? : ");
            scanf("%s", year);                      // stdin으로 연도 입력받음
            if (!strcmp(year, "0")) break;
            else {
                fp = fopen("type3.txt", "r");       // txt 파일에 저장된 query 읽어옴
                str = (char*)malloc(500 * sizeof(char));
                tmp = (char*)malloc(100 * sizeof(char));
                fgets(str, 500, fp);                // 입력받은 연도를 query에 넣어주는 과정
                strcat(str, year);
                fgets(tmp, 500, fp);
                strcat(str, tmp);
                printf("Customer name : ");
                state = mysql_query(connection, str);
                if (state != 0) continue;
                sql_result = mysql_store_result(connection);
                while ((sql_row = mysql_fetch_row(sql_result)) != NULL)
                {
                    printf("%s %s  ", sql_row[0], sql_row[1]);  // customer는 name, phone_number로 구분되므로 둘 다 표시
                }
                printf("₩n₩n");
                mysql_free_result(sql_result);
                free(str);
                fclose(fp);
            }
        }
    }
}
```

```
        else if (type == 4) {
            printf("---- TYPE IV ----₩n₩n");
            printf("** Find those packages that were not delivered within the promised time **₩n");
            fp = fopen("type4.txt", "r");       // txt 파일에 저장된 query 읽어옴
            while (!feof(fp)) {
                str = (char*)malloc(1400 * sizeof(char));
                fgets(str, 1400, fp);
                state = mysql_query(connection, str);
                if (state != 0) continue;
                sql_result = mysql_store_result(connection);
                while ((sql_row = mysql_fetch_row(sql_result)) != NULL)
                {
                    printf("%s  ", sql_row[0]); // package는 package_id로 구분되므로 package_id만 표시
                }
                printf("₩n₩n");
                mysql_free_result(sql_result);
                free(str);
            }
            fclose(fp);
        }
        else if (type == 5) {
            char day[8];
            char* y, * m;
            y = (char*)calloc(5, sizeof(char));
            m = (char*)calloc(3, sizeof(char));
            char* number = (char*)calloc(12, sizeof(char));
            char* fname = (char*)calloc(40, sizeof(char));
            printf("---- TYPE V ----₩n₩n");
            printf("** Generate the bill for each customer for the past certain month **₩n");
            printf("Customer name : ");
            scanf("%s", name);                  // stdin으로 customer 이름 입력받음
            printf("Customer number : ");
            scanf("%s", number);                // stdin으로 customer phone_number 입력받음 ( customer는 name, phone_number로 구분되므로 둘 다 입력)
            printf("Which month(YYYY-MM)? : ");
            scanf("%s", day);                   // stdin으로 날짜 입력받음
            strncpy(y, day, 4);
            strncpy(m, day + 5, 2);
            printf("Generating Bill...₩n₩n");

            strcat(fname, "bill_");             // output file 이름 생성 ("bill_YYYYMM_customer-name_customer-phone-number.txt")
            strcat(fname, y);
            strcat(fname, m);
            strcat(fname, "_");
            strcat(fname, name);
            strcat(fname, "_");
            strcat(fname, number);
            strcat(fname, ".txt");
            FILE* f_out = fopen(fname, "w");

            fp = fopen("type5-1.txt", "r");     // txt 파일에 저장된 query 읽어옴. type5-1.txt는 V의 첫 번째 항목에 해당하는 bill 생성
            str = (char*)malloc(500 * sizeof(char));
            tmp = (char*)malloc(200 * sizeof(char));
            fgets(str, 500, fp);                // 입력받은 customer name, phone_number, 날짜를 query에 넣어주는 과정
            strcat(str, "'");
            strcat(str, name);
            strcat(str, "'");
            fgets(tmp, 200, fp);
            strcat(str, tmp);
            strcat(str, "'");
            strcat(str, number);
            strcat(str, "'");
            fgets(tmp, 200, fp);
            strcat(str, tmp);
            strncat(str, y, 4);
            fgets(tmp, 200, fp);
            strcat(str, tmp);
            strncat(str, m, 2);
            state = mysql_query(connection, str);
            if (state != 0) continue;
            sql_result = mysql_store_result(connection);
            fprintf(f_out, "Customer₩tPhone_number₩tAddress₩tAmount₩n");
            while ((sql_row = mysql_fetch_row(sql_result)) != NULL)
            {
                fprintf(f_out, "%s₩t%s₩t%s₩t%s₩n", sql_row[0], sql_row[1], sql_row[2], sql_row[3]);
                // customer는 name, phone_number로 구분되므로 파일에 둘 다 표시
                //  A simple bill: customer, address, and amount owed  항목 포함
            }
            fprintf(f_out, "₩n");
            mysql_free_result(sql_result);
            free(str);
            fclose(fp);

```

```
302                 fprintf(f_out, "----------------------------₩n");
303                 fprintf(f_out, "Itemized Billing List₩n");
304                 fp = fopen("type5-2.txt", "r");     // txt 파일에 저장된 query 읽어옴. type5-2.txt는 V의 두 번째, 세 번째 항목에 해당하는 bill 생성
305                 str = (char*)malloc(500 * sizeof(char));
306                 tmp = (char*)malloc(200 * sizeof(char));
307                 fgets(str, 500, fp);                // 입력받은 customer name, phone_number, 날짜를 query에 넣어주는 과정
308                 strcat(str, "₩"");
309                 strcat(str, name);
310                 strcat(str, "₩"");
311                 fgets(tmp, 200, fp);
312                 strcat(str, tmp);
313                 strcat(str, "₩"");
314                 strcat(str, number);
315                 strcat(str, "₩"");
316                 fgets(tmp, 100, fp);
317                 strcat(str, tmp);
318                 strncat(str, y, 4);
319                 fgets(tmp, 100, fp);
320                 strcat(str, tmp);
321                 strncat(str, m, 2);
322                 fgets(tmp, 200, fp);
323                 strcat(str, tmp);
324
325                 state = mysql_query(connection, str);
326                 if (state != 0) continue;
327                 sql_result = mysql_store_result(connection);
328                 fprintf(f_out, "| Package Number | Amount | Serivce Type | Payment Type | Timeliness of Delivery|₩n");
329                 while ((sql_row = mysql_fetch_row(sql_result)) != NULL)
330                 {
331                     fprintf(f_out, "%s₩t%s₩t%s₩t%s₩t%s₩n", sql_row[0], sql_row[1], sql_row[2], sql_row[3], sql_row[4]);
332                     // A bill listing charges by type of service  항목 포함
333                     // An itemize billing listing each individual shipment and the charges for it  항목 포함
334                 }
335                 fprintf(f_out, "₩n");
336                 mysql_free_result(sql_result);
337                 free(str);
338                 fclose(fp);
339                 fclose(f_out);
340                 printf("Generation Completed₩n₩n");
341             }
342             else if (type == 0) break;
343             else continue;
344         }
345
346         fp = fopen("db_clear.txt", "r");// db_clear.txt에는 table delete, drop 코드가 있음
347         while (!feof(fp)) {             // 한 줄 씩 읽어 와서 database 삭제 수행
348             str = (char*)malloc(500 * sizeof(char));
349             fgets(str, 500, fp);
350
351             state = mysql_query(connection, str);
352             if (state != 0) continue;
353             sql_result = mysql_store_result(connection);
354             mysql_free_result(sql_result);
355             free(str);
356         }
357         fclose(fp);
358
359         mysql_close(connection);
360     }
```