Database System project #1

서강대학교 컴퓨터공학과

201216365 장종석

1. Entity & Attributes 생성 이유

ER 다이어그램이나 logical, physical 모드 요구사항 지정 및 설명에 앞서 이 부분을 설명하는 이유는 개인적인 생각에 의해 Entity와 Attributes를 생성했으므로 윗 부분을 설명하기 전에 Entity와 Attributes 생성 이유를 설명하기 위함이다.

1.1 Staff

* Entity 생성 이유

서강 배달의 직원들을 저장하기 위한 Entity가 필요하다.

* Attributes 생성 이유

Staff객체는 Staff\_Code와, Name, Age, gender, Address, work, telephone, birthday, salary, account 가 있어야 한다. 여기서 Staff\_Code의 경우 primary\_key로 사용한다.

1.2 User

* Entity 생성 이유

배달시킨 고객의 정보를 포함해야 하기 때문에, 회원정보를 수정, 조회를 해야 하므로,

고객정보를 나타내는 User 객체를 생성하였다.

* Attributes 생성 이유

User 객체는 User\_code, Name, Address, Telephone, E-mail 로 구성되어 있다. 해당

정보를 포함하라 요구하였기 때문에 그에 맞게 Attributes를 설정하였다.

User\_Code의 경우 primary\_key로 사용한다.

1.3 Store

* Entity 생성 이유

가맹점에 대한 정보를 포함하고, 해당 정보를 수정, 조회 해야하므로, 가맹점의 정보 를 나타내는 Store 객체를 생성하였다.

* Attributes 생성 이유

가맹점의 객체는 Store\_code, name, address, telephone, work\_time, menu를 정보를

가지고 있어야 하기 때문에 해당 값을 가지게 되었다.

store\_Code의 경우 primary\_key로 사용한다.

1.4 Rider

* Entity 생성 이유

프로젝트 요구사항에 따르면 서강 배달의 경우 라이더가 품목을 배달을 해줘야

하기때문에, Rider 객체를 생성하였다.

* Attributes 생성 이유

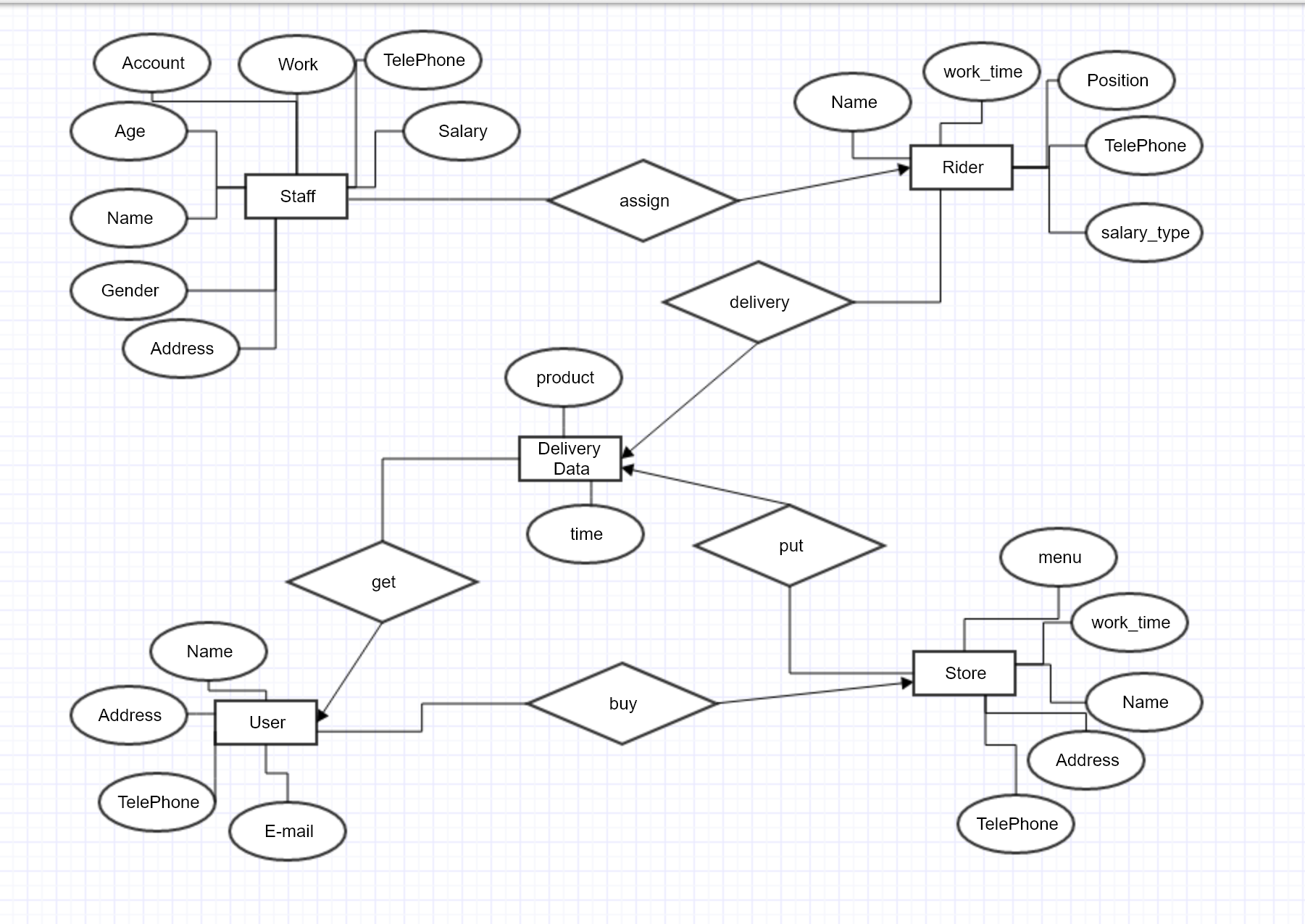
Rider객체는 Rider\_code, Name, work\_time, position, telephone, salary\_type의

정보를 가지고 있다. Rider\_code의 경우 primary\_key로 사용한다.

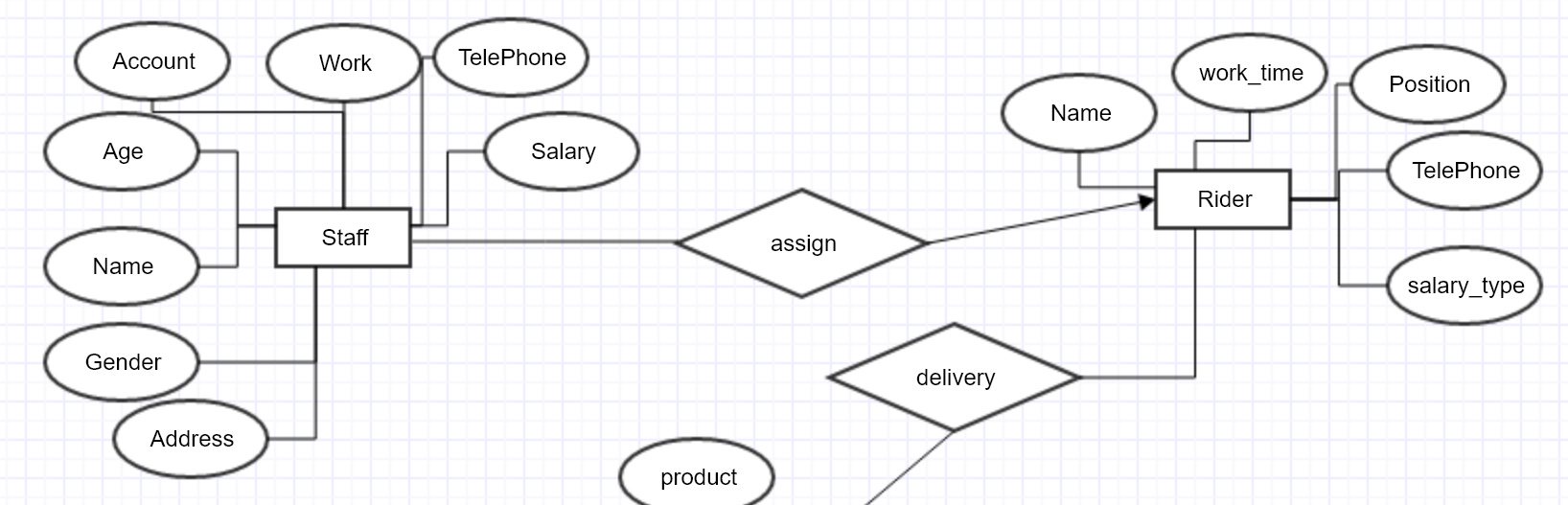
2. ER 다이어그램

위의 Entity를 이용해서 만든 전체 ER 다이어그램을 제시한 뒤 각각의 관계에 대해 설명하겠다.

2.1 전체 ER-다이어그램



2.2 Staff to Rider ( Assign Relation)

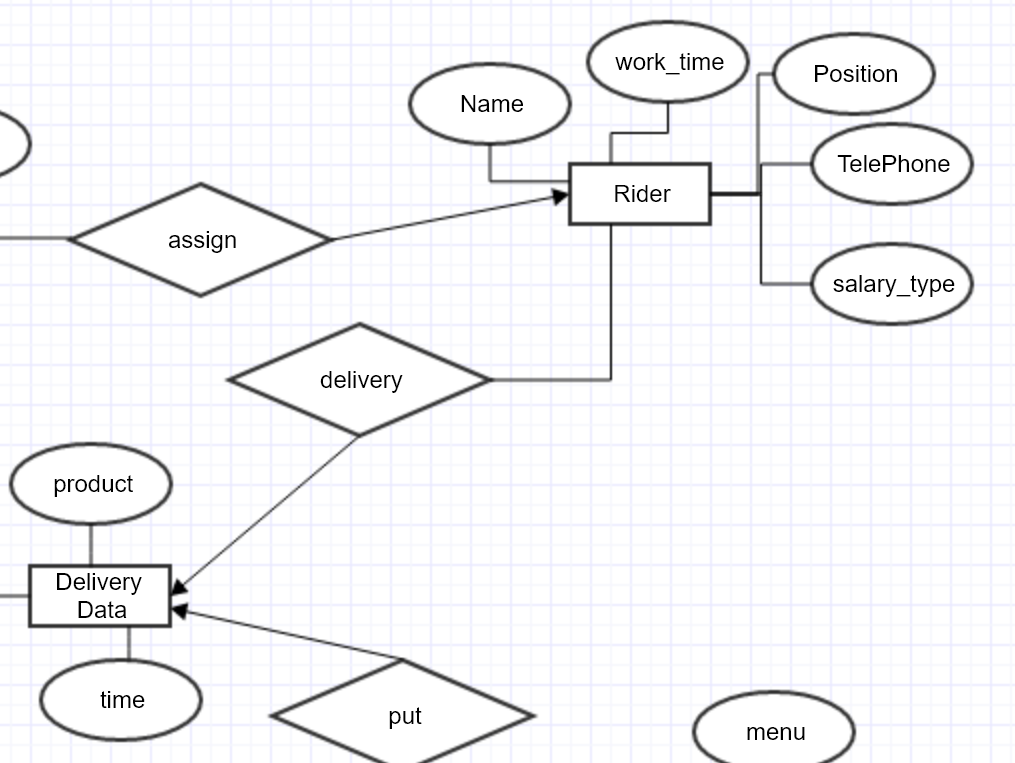


Relation Name ) Assign

Cardinality ) 1 : N Relationship

위의 Diagram은 Staff와 Rider사이의 ER diagram이다. Relation 이름은 Assign 으로 Staff가 Rider를 할당해주기 때문에 위와 같이 설정했다. 한명의 Staff가 여러명의 Rider를 매칭하기 때문에 1 : N Cardinality 를 갖는다. Telephone 의 경우 두개 이상이 가질 수 있지만, 회사에서 업무용으로 사용하는 전화번호의 경우 하나의 정보를 입력하는 것이 서로간의 통신이 원할하기 때문에 하나를 입력하였다.

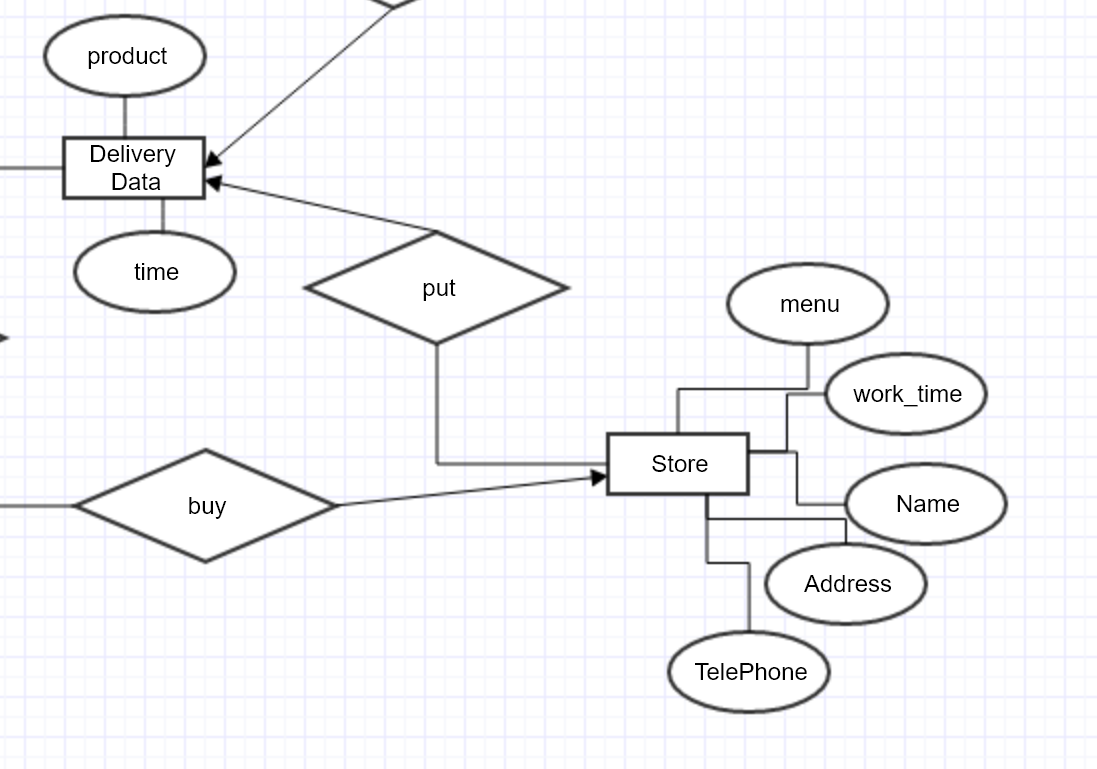
2.3 Rider to Delivery Data (delivery relationship)



Relation Name ) delivery

Cardinality ) 1 : N Relationship

위의 Diagram은 Rider와 Delivery Data사이의 ER-diagram 이다. Relation 이름은 delivery로 한명의 Rider가 여러개의 delivery Data를 배송할 수 있기 때문에 delivery 라는 이름을 설정했으며, 1:N Relationship 으로 설정하였다.

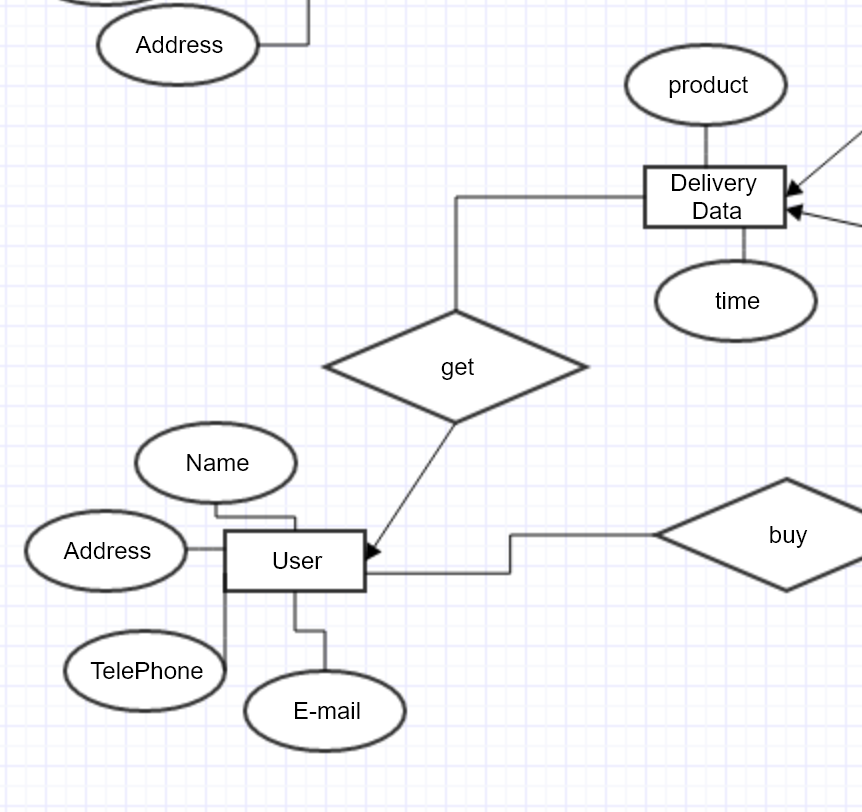
2.3 store to delivery Data (put relationship)

Relation Name ) put

Cardinality ) 1 : N Relationship

위의 Diagram은 Store와 Delivery Data사이의 ER-diagram 이다. relation 이름은 put으로 정했는데, store 가 delivery data를 제공하기 때문이다. 하나의 Delivery data에는 여러 주문이 들어 있을 수 있기 때문에 1: N Relationship 으로 설정하였다.

2.4 user to DeliveryData ( get relationship)

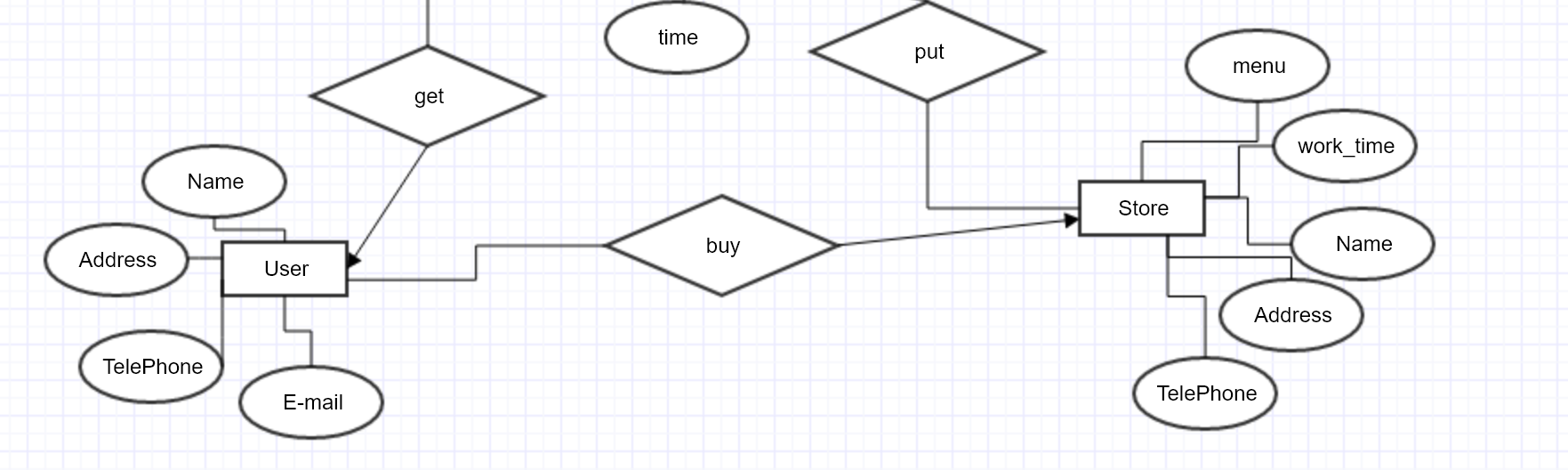


Relation Name ) get

Cardinality ) 1 : N Relationship

위의 Diagram은 user와 Delivery Data사이의 ER-diagram 이다. relation 이름은 get으로 정했는데, user가 delivery data를 제공받기 때문이다. 하나의 Delivery data에는 여러 주문이 들어 있을 수 있기 때문에 1: N Relationship 으로 설정하였다.

2.5 user to store( buy relationship)



Relation Name ) buy

Cardinality ) 1 : N Relationship

위의 Diagram은 user와 store사이의 ER-diagram 이다. relation 이름은 call으로 정했는데, user가 전화를 통해서 Store에 주문을 하기 때문이다.

여러 사용자가 하나의 매장에 전화를 하여 주문을 할 수 있기 때문에 1: N Relationship 으로 설정하였다.

3. Logical Mode

위의 Entity & Attributes 생성 이유에서 Entity와 Attribute 생성 이유에 대해 기술 했으므로, 3 Logical Mode에서는 Logical 모드에서의 Entity의 모습과 Entity의 Attributes 표를 제시하고 Entity 사이의 관계를 설명하겠다.

3.1 Entity

* Staff

|  |  |
| --- | --- |
| Attributes | Feature |
|  | Staff\_code : Staff primary 값으로 고유한 값을 가진다 |
| Name: Staff 의 이름을 저장한다. |
| Age : Staff의 나이를 나타낸다. |
| Gender : Staff의 성별을 나타낸다. |
| Address : Staff의 주소를 저장한다. |
| Work : 담당업무를 나타낸다. |
| Telephone : 전화번호를 나타낸다. |
| birthDays : 생년월일을 나타낸다. |
| Salary : 급여정보를 나타낸다. |
| Account : 급여 계좌를 나타낸다. |

* Store

|  |  |
| --- | --- |
| Attributes | Feature |
|  | store\_code : 가맹점 고유 코드를 나타낸다 |
| name : 가맹점 이름을 나타낸다 |
| Address : 가맹점 주소를 저장한다 |
| telephone : 가맹점 전화번호를 나타낸다. |
| work\_time : 가맹점 영업시간을 나타낸다 |
| menu : 가맹점 음식 메뉴를 나타낸다. |

* User

|  |  |
| --- | --- |
| Attributes | Feature |
|  | user\_code : 사용자 고유 코드를 나타낸다 |
| Name : 사용자 이름을 저장하한 |
| Address : 사용자 주소를 저장한다 |
| Telephone : 사용자 전화번호를 저장한다 |
| E-mail : 사용자 이메일을 나타낸다. |

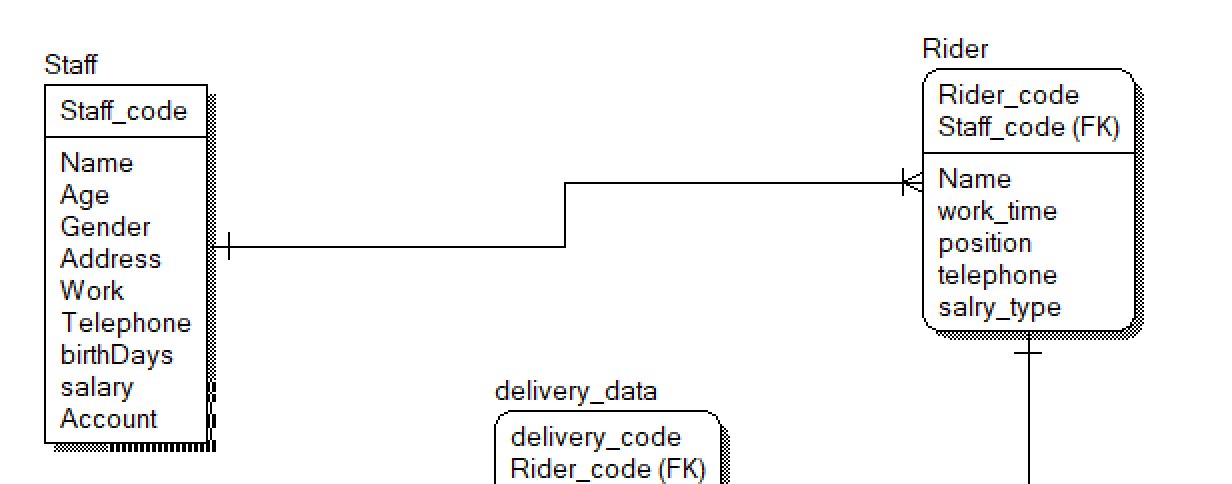
* Rider

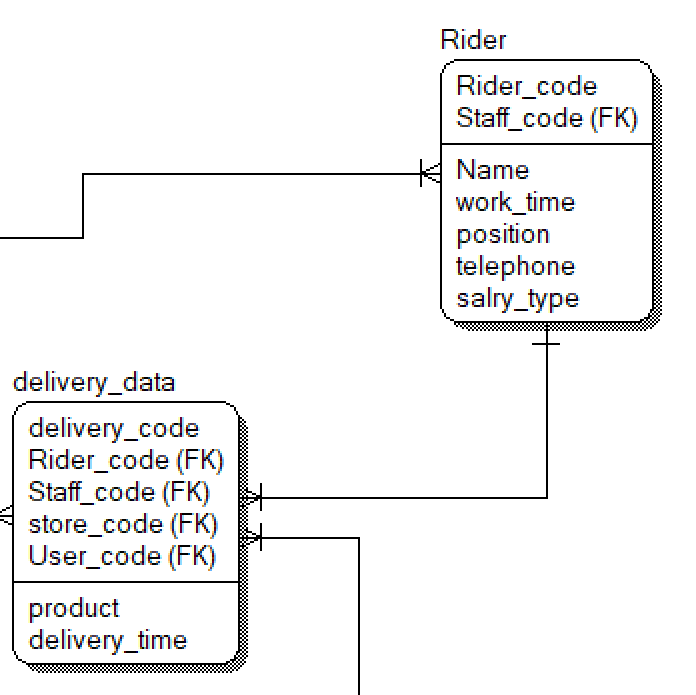
|  |  |
| --- | --- |
| Attributes | Feature |
|  | Rider\_code : 라이더 코드를 나타낸다 |
| Name : 라이더 이름을 저장한다 |
| Work\_time : 근무시간을 나타낸다 |
| position : 현재 위치를 좌표로 나타낸다 |
| telephone : 라이더의 전화번호를 저장한다 |
| salry\_type : 급여형태를 나타낸다. |

* Delivery\_data

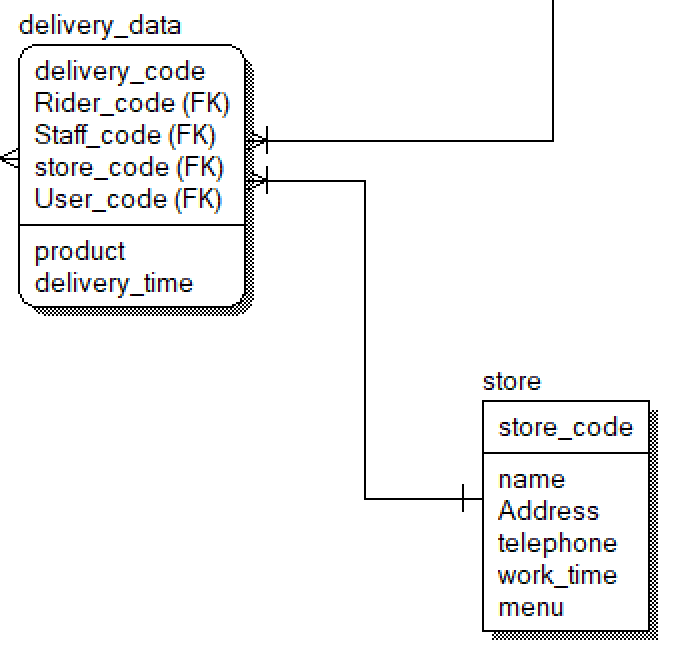
|  |  |
| --- | --- |
| Attributes | Feature |
|  | delivery\_code : 주문 고유번호를 나타낸다 |
| product : 메뉴의 품목을 나타낸다 |
| delivery\_time : 배달 시간을 나타낸다. |
|

3.2 Relation

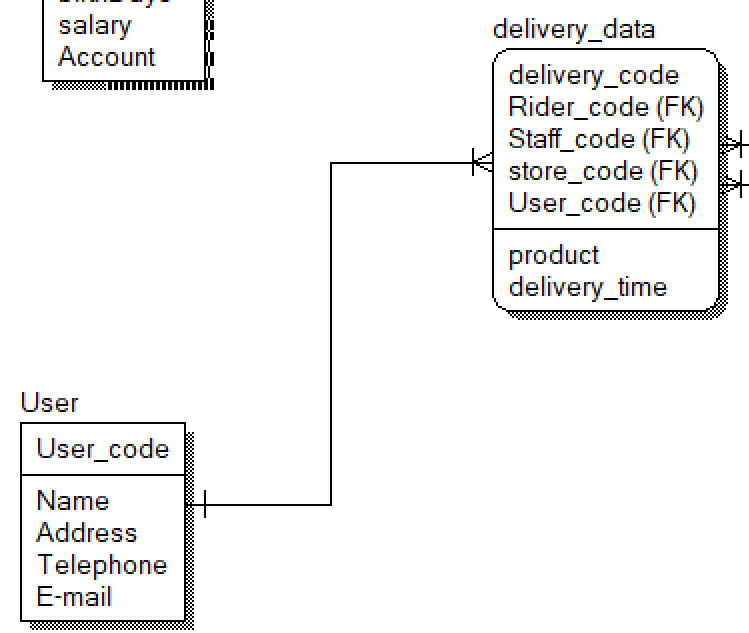
* Staff to Rider
  + cardinality : One or More
  + Relation Type : Identifying
  + Staff 가 배달 라이더를 배정한다고 명세서에 나타나 있었다. 즉, 스태프와 라이더간의 릴레이션이 존재하는데, 이는 한명의 스태프가 여러명의 라이더를 관리 할 수 있다는 의미가 되므로, one or more Cardinality 이다.
* Rider to Delivery\_data



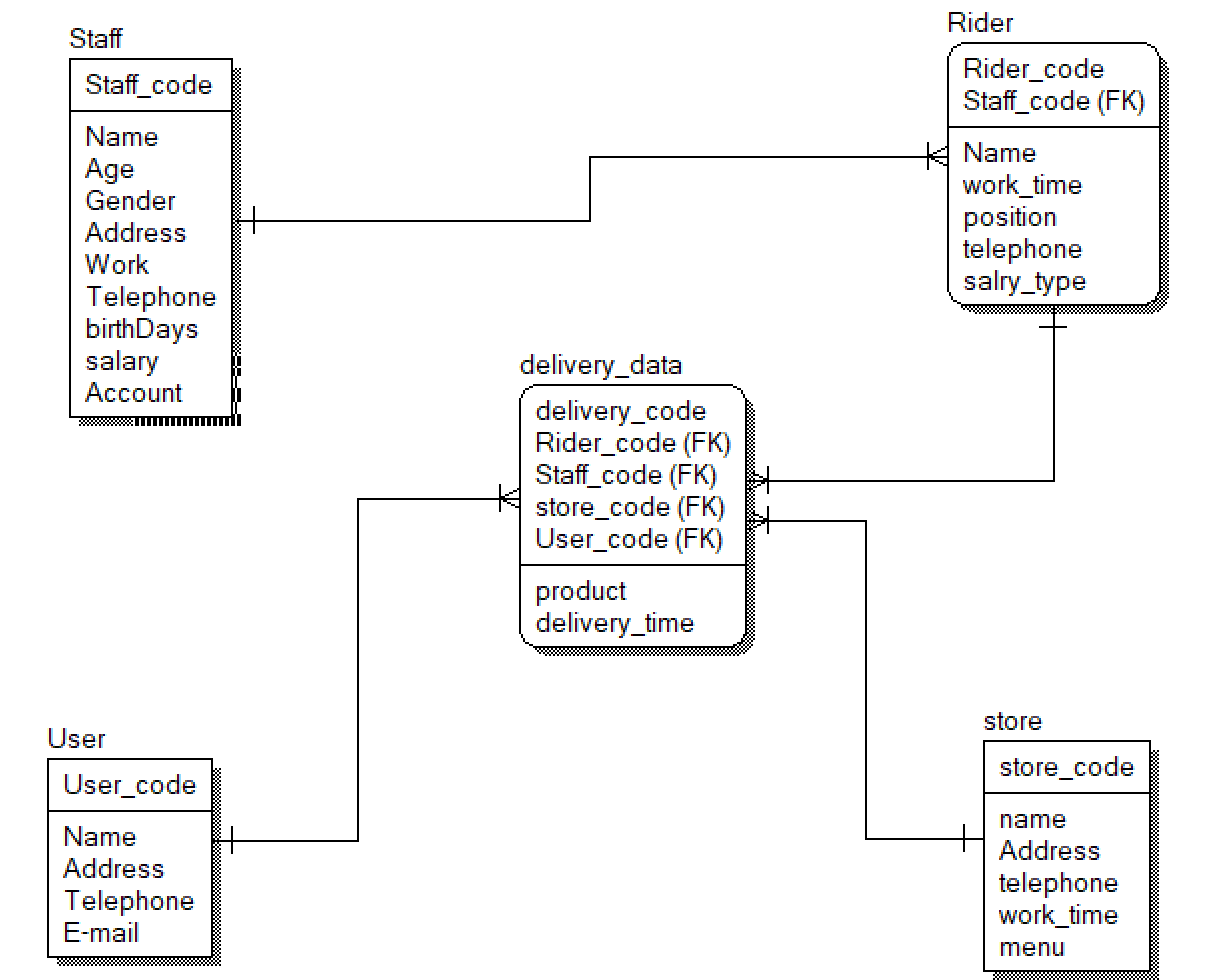
* + cardinality : One or More
  + Relation Type : Identifying
  + Rider는 여러 배달을 진행할 수 있다. 따라서 한명의 배달직원이 여러 배달정보를 가져야 할기 때문에 Cardinality의 경우 one, more로 설정을 하였으며, 이 경우 배달에 한정되기 때문에 라이더가 배정되지 않을 수 없다.
* store to deliver data



* + cardinality : One or More
  + Relation Type : Identifying
  + 한 체인점에서 여러 배달정보를 생성할 수 있다. 이를 사용하게 되면 Cardinality가 one or more로 설정됨을 알 수 있으며, 두 관계에 Null이 되는 경우는 절대로 발생하지 않기 때문에 Identifying 으로 설정하였다
* user to Delviery\_data



* + cardinality : One or More
  + Relation Type : Identifying
  + 한 유저의 경우 여러번의 배달을 주문할수 있기때문에 한 명의 유저가 여러개의 여러개의 배달정보를 가져올 수 있기때문에 이 또한 one or more Cardinality 로 설정을 하였고, 유저정보가 없는사람이 배달을 할 수 없기 때문에 Identifying 또한 Null 이 될 수 없다.
* 전체 구현결과



4. Physical Mode

4.1 Domain & Constraint

* **Staff\_code, Rider\_code, User\_code, store\_code, delivery\_code**

의 Domain은 String이 Domain parent인 Domain 이다. 이 Domain들은 Constraint는

없지만 VARCHAR(32)로 하여 32자리로 제한을 두었다.

* **Address, Name, telephone, E-mail, Work, Account, menu**

위의 Domain은 String이 Domain parent인 Domain 이다. 이들 역시 Constaint는 없지만 VARCHAR()에서 ()안에 수의 제한을 두었다. 예를들어서 Address 의 경우 VARCHAR(128) 처럼 하여 128 까지의 문자열의 길이 제한을 두었다.

* **work\_time, birthday**

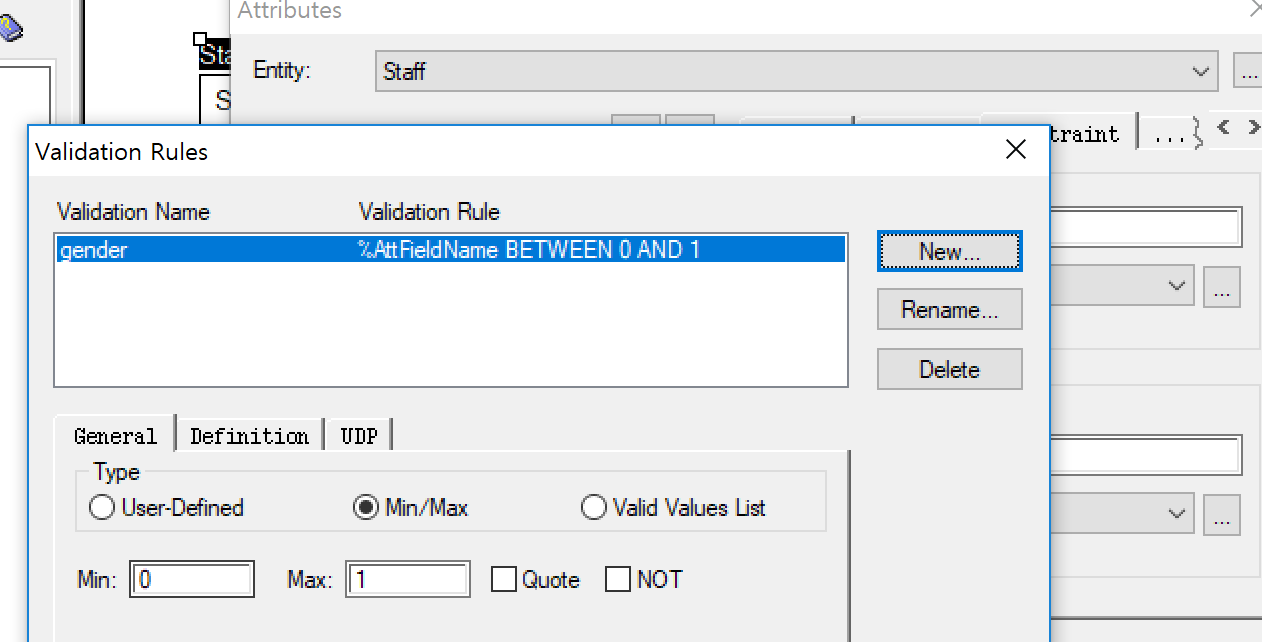
work\_time, birthday 의 경우 DateTime이 Domain Parent인 Domain이다. 영업시간의

경우 시간으로 표기를 할 수 있기때문에 DateTime을 사용하였다.

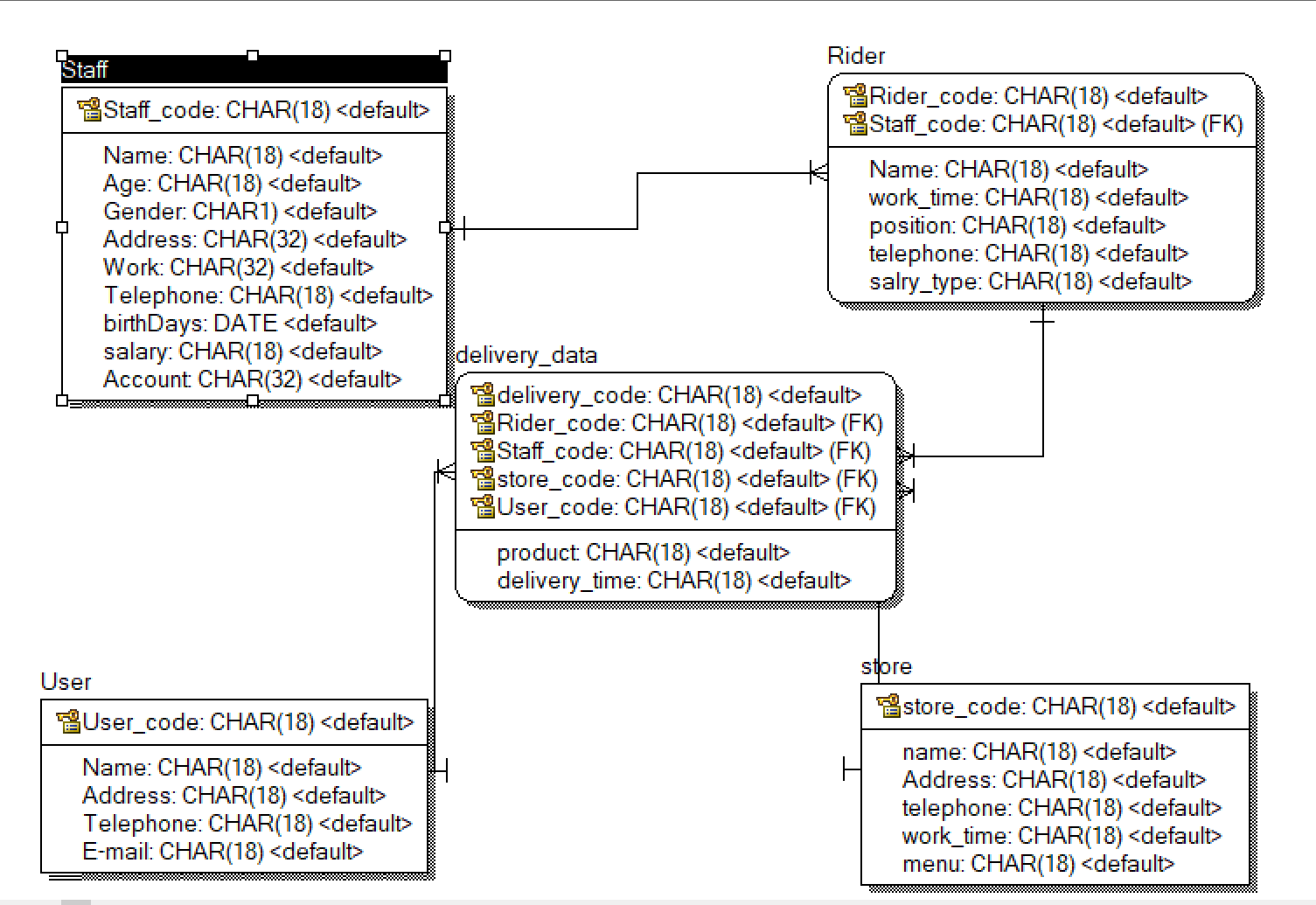
* Gender

Gender의 경우 0, 1 두가지로만 표현을 하도록 하였는데, 이 경우 하나는 남성, 하나는

여성을 나타내는 식별자로서 사용하도록 하였다.



4.2 구현 결과



5. 모든 설계에 대한 세부 요구사항 지정 및 설명

5.1) 업무 관리 시스템 어플리케이션

* 업무 관리 시스템에서는 직원 관리, 회원 관리, 가맹점 관리, 라이더 관리, 배달 내역

관리를 위한 메뉴가 존재하며 이를 이용해 업무 절차의 완전한 자동화를 위한 데이

터베이스 설계가 필요하다. 데이터베이스 설계자는 다음의 어플리케이션 프로토타입

에 적당한 데이터베이스를 모델링하고 사용자와 함께 검토해야 한다.

=>Staff, Store, user, rider, delivery\_data entity 를 이용하면 해당 관리 시스템을

구현할 수 있다.

5.2) 직원 관리 시스템 어플리케이션

* 서강 배달에서는 직원 관리를 위해 인트라넷 기반 직원 정보 시스템을 운영한다. 경

영 담당자는 모든 직원의 정보에 대한 삽입/삭제/수정/조회 기능을 수행할 수 있다.

* 직원 정보를 조회할 경우 아래와 같은 화면에서 각 직원에 대한 개인정보를 포함한

신상 정보와 담당 분야와 업무에 관련된 모든 정보를 확인할 수 있다.

=> 직원 관리의 경우 Staff 객체를 이용하여 구현하였다. Staff에 데이터를 입력하고, 삭제하고 하는 것으로 해당 데이터를 수정할 수 있다.

5.3) 회원 관리 시스템 어플리케이션

* 새로운 회원이 배달을 처음 의뢰할 경우 경영자는 회원의 개인정보를 먼저 입력하

고 고객의 개인정보 입력 창은 다음과 같이 구성된다.

=>회원관리의 경우 User 객체를 이용하여 구현하였는데, 회원들이 개인정보를

입력하면 해당 데이터를 저장할 수 있기 때문에, 위를 구현하는데 문제가 없었다.

5.4) 가맹점 관리 시스템 어플리케이션

* 서강 배달 경영자는 배달 가능한 업체들과 제휴를 맺고 가맹점들의 정보를

관리한다. 경영자는 모든 가맹점들의 정보에 대한 삽입/삭제/수정/조회 기능을

수행할 수 있다.

* 가맹점 정보를 조회할 경우 아래와 같은 화면에서 각 가맹점에 대한 모든 정보를 확

인할 수 있다.

=>가맹점 관리의 경우 Sotre 객체를 이용하여 만들었는데, Store 객체에 포함되어

있는 내용들을 조회, 삭제, 수정 등등을 통해서 원하는 기능을 구현하였으며, 모든 정보를 포함한다.

5.5) 라이더 정보 관리 시스템 어플리케이션

* 배달을 실행하는 라이더들에게 효율적으로 일을 분담하기 위해 라이더 정보를 저장

해야한다.

=>라이더의 정보를 관리하기 위하여 Rider 객체를 사용하였는데, 해당 객체에는 라이더에 대한 정보들이 포함되어 있기때문에 원하는 요구사항을 충족 시킬 수 있다.

5.6) 배달 내역 관리 시스템 어플리케

* 배달 업무 관리의 효율성을 위해 정보 시스템에서는 아래와 같이 가맹점 고객 및

라이더의 배달 내역 정보를 저장한다.

Delivery\_data를 통하여 해당 부분을 구현하였으며, 여기에 가맹점, 유저, 라이더

정보등을 외부키로 사용하였기 때문에 해당 정보들을 사용할 수 있다.

6.7) 서강 배달에서 사용되는 정보 시스템의 요구사항

* 서강 배달에서 개발하려는 새로운 정보 시스템에서는 음식 배달에 대한 배달 현황

을 관리하고 배달를 의뢰한 고객과 직원 및 라이더의 정보를 관리할 수 있어야 하

며, 가맹점에 대한 정보와 매출 정보 관리가 필요하다.

=> 해당 시스템을 구현하기 위하여 위와 같은 구현사항들을 통하여 시스템을 구현함

* 직원 정보: 서강 배달의 직원은 의뢰받은 배달에 대해 라이더를 배정하거나 재무 담

당, 홍보 및 홈페이지 관리 등을 진행한다. 경영진은 업무 분담과 급여관리 등 직원

관리의 수월성을 위해 모든 직원의 개인 정보를 기록, 관리하려고 한다. 직원의 개

인 정보는 이름, 나이, 성별, 주소, 담당 업무, 전화번호, 생년월일, 연봉, 급여계좌를

포함한다. 직원의 주소는 거주지에 대한 시, 군, 구, 동 등에 대한 주소 정보와 우편

번호 정보를 포함한다.

=> 직원(Staff)에 관련된 데이터를 처리하기 위하여 Staff 객체를 생성하여 해당 데이터

를 저장할 수 있도록 구현하였다.

* 회원 정보: 서강 배달은 효율적인 업무 진행을 위해 직원 정보와 마찬가지로 배달을

시킨 회원 정보를 저장 관리한다. 개별 회원 정보는 다음과 같이 이름, 주소, 전화번

호, 이메일을 포함한다.

=> 회원 정보또한 위에서 설명한 것과같이 구현을 하였고, 이를 통하여 회원정보

조회, 수정, 삭제, 입력 등이 가능하도록 하였다.

* 가맹점 정보: 서강 배달은 가맹점들에 대한 정보를 저장 관리한다. 가맹점 정보는

다음과 같이 이름, 주소, 전화번호, 영업 시간, 메뉴를 포함한다.

=> 가맹점에 대한 정보를 나타내기 위하여 Store라는 객체를 구성하여 해당 데이터

가 저장될 수 있도록 시스템을 구현하였다.

* 라이더 정보: 서강 배달은 원활한 배달을 위해 라이더들의 정보를 저장 관리한다.

라이더 정보는 다음과 같이 이름, 근무 시간, 현재 위치, 전화번호, 급여 유형을 포

함한다.

=> 라이더에 대한 정보를 구현하기 위하여 위와 같은 방법으로 Rdier 객체를 만들어

필요한 데이터가 저장될 수 있도록 시스템을 구현하였다.

* 배달 내역 정보: 서강 배달은 효율적인 배달을 위해 각 회원, 가맹점, 라이더별 배달

내역을 기록, 관리하려고 한다. 배달 내역에는 배달을 시킨 고객, 배달될 음식을 만

드는 가맹점, 배달을 하는 라이더, 가격, 배달 품목, 배달 시간을 포함한다.

=> 배달정보의 경우 라이더, 유저, 가맹점의 정보가 모두 포함되어야 하기때문에

위와 같이 외부키를 연결하여 사용하도록 시스템을 구현하였다.