Project 2

Report

이름: 이예준

학번: 20212022

**1. BCNF Decomposition**

-특정 Schema가 BCNF를 만족시키기 위해서는 해당 Schema의 모든

Functional Dependency α → β에 대하여

**1. α가 superkey 이다.**

**2. trivial한 α → β가 존재한다.**

를 확인한다.

만약 그런 α → β가 존재하면 해당 Schema는 BCNF를 만족하지만,

존재하지 않는다면 해당 Schema는 BCNF를 만족하기 위한 Decomposition이 필요하다.

기존 Schema가 R이고, R이 BCNF를 위반하게 만드는 Functional Dependency가

α → β일 때, R을 **( α ∪ β )**와 **( R – ( β – α ) )**로 분해한 뒤 BCNF를 만족하는지 확인한다.

이 과정은 BCNF를 만족할 때까지 반복한다.

**1. Agent**

- Attributes: Agent\_id (PK), Name, Address, Email, Phone\_num

- FDs:

- Agent\_id → Name, Address, Email, Phone\_num

\* Agent\_id가 primary key이므로 BCNF를 만족한다.

**2. Agent\_Market**

- Attributes: Market\_id (FK), Agent\_id (FK)

- FDs:

- Market\_id, Agent\_id → (Market\_id, Agent\_id)

\* Market\_id, Agent\_id가 composite priamry key이고, trivial하므로 BCNF를 만족한다.

**3. Agent\_Sold**

- Attributes: Agent\_id (FK), Sold\_id (FK), Buy, Sell

- FDs:

- Agent\_id, Sold\_id → Buy, Sell

\* Agent\_id, Sold\_id가 composite priamry key이므로 BCNF를 만족한다.

**4. Buyer**

- Attributes: Buyer\_id (PK), Name, Address, Email, Phone\_num

- FDs:

- Buyer\_id → Name, Address, Email, Phone\_num

\* Buyer\_id가 primary key이므로 BCNF를 만족한다.

**5. Buyer\_Sold**

- Attributes: Sold\_id (FK), Buyer\_id (FK)

- FDs:

- Sold\_id, Buyer\_id → (Sold\_id, Buyer\_id)

\* Sold\_id, Buyer\_id가 composite priamry key이므로 BCNF를 만족한다.

**6. Market**

- Attributes: Market\_id (PK), Price, Name, Address, Enroll\_date, School\_district, Type

- FDs:

- Market\_id → Price, Name, Address, Enroll\_date, School\_district, Type

\* Market\_id가 primary key이므로 BCNF를 만족한다.

**7. Market\_Photo**

- Attributes: Market\_id (FK), Photo\_id (FK)

- FDs:

- Market\_id, Photo\_id → (Market\_id, Photo\_id)

\* Market\_id, Photo\_id가 composite priamry key이므로 BCNF를 만족한다.

**8. Market\_Room**

- Attributes: Market\_id (FK), Room\_id (FK)

- FDs:

- Market\_id, Room\_id → (Market\_id, Room\_id)

\* Market\_id, Room\_id가 composite priamry key이므로 BCNF를 만족한다.

**9. Photo**

- Attributes: Photo\_id (PK), Interior\_photo, Exterior\_photo, Inspection\_photo

- FDs:

- Photo\_id → Interior\_photo, Exterior\_photo, Inspection\_photo

\* Photo\_id가 primary key이므로 BCNF를 만족한다.

**10. Room**

- Attributes: Room\_id (PK), Bed\_room, Bath\_room

- FDs:

- Room\_id → Bed\_room, Bath\_room

\* Room\_id가 primary key이므로 BCNF를 만족한다.

**11. Seller**

- Attributes: Seller\_id (PK), Name, Address, Phone\_num

- FDs:

- Seller\_id → Name, Address, Phone\_num

\* Seller\_id가 primary key이므로 BCNF를 만족한다.

**12. Seller\_Market**

- Attributes: Seller\_id (FK), Market\_id (FK)

- FDs:

- {Seller\_id, Market\_id} is the composite primary key/

- Seller\_id, Market\_id → (Seller\_id, Market\_id)

\* Seller\_id, Market\_id가 composite priamry key이므로 BCNF를 만족한다.

**13. Seller\_Sold**

- Attributes: Seller\_id (FK), Sold\_id (FK)

- FDs:

- Seller\_id, Sold\_id → (Seller\_id, Sold\_id)

\* Seller\_id, Sold\_id가 composite priamry key이므로 BCNF를 만족한다.

**14. Sold**

- Attributes: Sold\_id (PK), Price, Name, Address, Sold\_date, School\_district, Type

- FDs:

- Sold\_id → Price, Name, Address, Sold\_date, School\_district, Type

\* Sold\_id가 primary key이므로 BCNF를 만족한다.

**15. Sold\_Photo**

- Attributes: Photo\_id (FK), Sold\_id (FK)

- FDs:

- {Photo\_id, Sold\_id} is the composite primary key, so:

- Photo\_id, Sold\_id → (Photo\_id, Sold\_id)

\* Photo\_id, Sold\_id가 composite priamry key이므로 BCNF를 만족한다.

**\* Summary**

모든 Schema가 BCNF를 만족하며, 그 이유는 모든 FD의 LHS에는 candidate key 또는

primary key의 일부가 있고, partial 또는 ransitive dependency가 있기 때문이다.

모든 Decomposition을 마친 Logical Schema Diagram은 아래의 그림과 같다.

텍스트, 도표, 라인, 폰트이(가) 표시된 사진

자동 생성된 설명

**2. Physical Schema Diagram**

위에서 Decomposition을 마친 Schema들에 대하여 각 스키마에 대한 Data Type,

Doamin Constraint, Relationship Type 및 NULL 허용 여부를 정의한다.

**1. Agent**

- Agent\_id: INTEGER (PK, NOT NULL)

- Name: VARCHAR(50), NOT NULL

- Address: VARCHAR(100), NOT NULL

- Email: VARCHAR(100), NOT NULL

- Phone\_num: VARCHAR(13), NOT NULL

**2. Agent\_Market**

- Market\_id: INTEGER (FK, NOT NULL)

- Agent\_id: INTEGER (FK, NOT NULL)

**3. Agent\_Sold**

- Agent\_id: INTEGER (FK, NOT NULL)

- Sold\_id: INTEGER (FK, NOT NULL)

- Buy: VARCHAR(1), NOT NULL

- Sell: VARCHAR(1), NOT NULL

**4. Buyer**

- Buyer\_id: INTEGER (PK, NOT NULL)

- Name: VARCHAR(50), NOT NULL

- Address: VARCHAR(100), NOT NULL

- Email: VARCHAR(100), NOT NULL

- Phone\_num: VARCHAR(13), NOT NULL

**5. Buyer\_Sold**

- Sold\_id: INTEGER (FK, NOT NULL)

- Buyer\_id: INTEGER (FK, NOT NULL)

**6. Market**

- Market\_id: INTEGER (PK, NOT NULL)

- Price: INTEGER, NOT NULL

- Name: VARCHAR(50), NOT NULL

- Address: VARCHAR(100), NOT NULL

- Enroll\_date: INTEGER, NOT NULL

- School\_district: INTEGER, NOT NULL

- Type: VARCHAR(20), NOT NULL

**7. Market\_Photo**

- Market\_id: INTEGER (FK, NOT NULL)

- Photo\_id: INTEGER (FK, NOT NULL)

**8. Market\_Room**

- Market\_id: INTEGER (FK, NOT NULL)

- Room\_id: INTEGER (FK, NOT NULL)

**9. Photo**

- Photo\_id: INTEGER (PK, NOT NULL)

- Interior\_photo: INTEGER, NOT NULL

- Exterior\_photo: INTEGER, NOT NULL

- Inspection\_photo: INTEGER, NOT NULL

**10. Room**

- Room\_id: INTEGER (PK, NOT NULL)

- Bed\_room: INTEGER, NOT NULL

- Bath\_room: INTEGER, NOT NULL

**11. Seller**

- Seller\_id: INTEGER (PK, NOT NULL)

- Name: VARCHAR(50), NOT NULL

- Address: VARCHAR(100), NOT NULL

- Email: VARCHAR(100), NOT NULL

- Phone\_num: VARCHAR(13), NOT NULL

**12. Seller\_Market**

- Seller\_id: INTEGER (FK, NOT NULL)

- Market\_id: INTEGER (FK, NOT NULL)

**13. Seller\_Sold**

- Seller\_id: INTEGER (FK, NOT NULL)

- Sold\_id: INTEGER (FK, NOT NULL)

**14. Sold**

- Sold\_id: INTEGER (PK, NOT NULL)

- Price: INTEGER, NOT NULL

- Name: VARCHAR(50), NOT NULL

- Address: VARCHAR(100), NOT NULL

- Sold\_date: INTEGER, NOT NULL

- School\_district: INTEGER, NOT NULL

- Type: VARCHAR(20), NOT NULL

**15. Sold\_Photo**

- Photo\_id: INTEGER (FK, NOT NULL)

- Sold\_id: INTEGER (FK, NOT NULL)

ID, 가격 및 개수와 같은 값들은 INTEGER Type으로 정의했으며,

이름 또는 주소와 같이 경우에 따라서 길이의 차이가 클 수 있는 경우에는

VARCHAR(50), VARCHAR(100)과 같이 가변 String Type으로 정의했다.

위에서 모든 정의를 마친 Physical Schema Diagram은 아래의 그림과 같다.

텍스트, 도표, 라인, 폰트이(가) 표시된 사진

자동 생성된 설명

\*Physical Schema Diagram

**3. Queries**

**1. CRUD File**

먼저, 데이터베이스 초기화를 위해 사용자로부터 쿼리를 입력받기 전에 텍스트 파일을 읽고

ODBC를 통해 데이터베이스에 테이블을 생성하고 데이터를 입력하는 과정이 필요하며,

사용자가 프로그램 종료를 요청하면 생성된 테이블을 모두 제거해야 한다.

이를 위해 두 개의 입력 파일 (20212022\_1.txt와 20212022\_2.txt)을 사용했다.

텍스트, 스크린샷, 라인, 패턴이(가) 표시된 사진

자동 생성된 설명

\*20212022-1.txt 일부분

텍스트, 폰트, 스크린샷, 타이포그래피이(가) 표시된 사진

자동 생성된 설명

\*20212022-2.txt

두 개의 텍스트 파일이 있는데, 첫 번째 파일인 20212022\_1.txt에는 테이블을 생성하고

데이터를 삽입하는 명령어가 포함되어 있다. 두 번째 파일인 20212022\_2.txt에는

20212022\_1.txt에서 생성된 테이블들을 제거하는 명령어가 포함되어 있다.

테이블을 제거할 때는 참조 관계를 고려하여 제거 순서를 유의해야 한다.

**2. Query**

쿼리를 처리하기 위해서는 서브쿼리를 포함하여 총 13개의 쿼리를 처리해야 한다.

먼저 텍스트 파일을 통해 데이터베이스에 데이터를 입력한 후, 사용자로부터 원하는 쿼리를

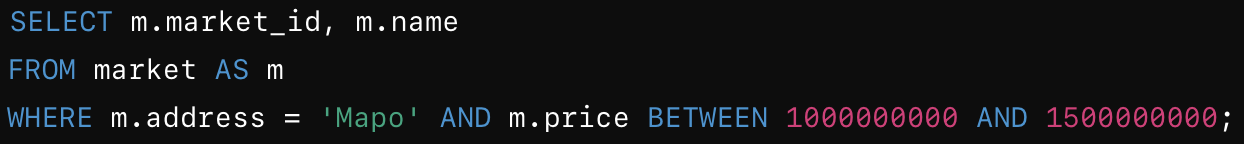
입력받는다.

**- TYPE 1**

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**- TYPE 1-1**



**- TYPE 2**

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

**- TYPE 2-1**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**- TYPE 3**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**- TYPE 3-1**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**- TYPE 3-2**

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

**- TYPE 4**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**- TYPE 4-1**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**- TYPE 4-2**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**- TYPE 5**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**- TYPE 6**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

**- TYPE 7**

