|  |
| --- |
| **2024년 2학기 데이터베이스 결과보고서** |

|  |  |
| --- | --- |
| **주**  **제** | 음악 방송 순위에 영향을 주는 각종 데이터들을 취합해 1위 가능성을 예측하고 가입된 회원들에게만 공유해 투표 전략 등을 논의하는 사이트를 위한 데이터베이스 설계. |
| **진**  **행**  **일**  **정**  **표** | - 결과 보고서 제출 전까지 전체 일정 포함   |  |  | | --- | --- | | 주차 | 진행 | | 5~7주차 | 팀 구성 | | 8~9주차 | 주제 탐색 및 정의, 사용 가능한 데이터 수집 | | 10주차 | ER 다이어그램 작성 | | 11주차 | 주제 변경, ER 다이어그램 초안 작성 | | 12주차 | 개념적 설계 과정 재확인 | | 13주차 | 논리적 설계 과정 확인, 데이터 무결성 및 정규화 과정, ER 다이어그램 수정 | | 14주차 | 테이블 생성 및 초기 데이터 입력 쿼리 작성 | | 15주차 | 통계 쿼리 작성, 결과보고서 작성 후 제출 | |
| **팀**  **원**  **기**  **여**  **도** | - 팀원 기여 내용 구체적으로 명시  - 유상욱 : 릴레이션 스키마 작성, 결과보고서 초안 작성, 데이터 수집 및 상세 편집, 쿼리 작성  - 오동근 : 데이터 수집 및 편집, 쿼리 작성  - 이은영 : 메인 주제 구성, E-R 다이어그램 작성, 릴레이션 스키마 작성 보조, 결과보고서 진행상황 작성.  - 이형민 : 데이터 탐색 및 수집, 데이터 생성 보조, 쿼리 오류 수정 |
| **현**  **재**  **까**  **지**  **진**  **행**  **상**  **황** | - 진행 사항 구체적으로 작성  **1. 주제 변경**  중간보고서 제출 후, 기존 주제가 지나치게 포괄적이라는 판단에 따라 주제를 수정하였습니다. 초기 설계는 소속사, 아티스트, 팬덤 모두를 포함하는 데이터베이스를 목표로 하였으나, 프로젝트의 범위를 축소하여 팬덤 위주의 데이터베이스로 초점을 맞추게 되었습니다.  음악 아티스트의 팬덤이 데이터베이스를 주로 활용하는 목적은 아티스트의 다양한 성적 지표를 확인하는 데 있습니다. 팬덤은 음원 순위, 음반 판매량 등 주요 성적 지표를 통해 아티스트의 인기도와 팬덤의 구매력을 파악하며, 특히 음악방송 순위에 영향을 미치는 데이터를 가장 중요하게 여깁니다.  음악방송은 아티스트의 스트리밍 수, 음반 판매량, 유튜브 조회수, 팬덤의 유료 앱 투표 점수 등을 종합적으로 평가하여 순위를 산정합니다. 팬덤은 이러한 데이터를 분석하여 예상 점수를 계산하고, 자신이 응원하는 아티스트(“최애”)가 1위를 차지하도록 전략을 세웁니다. 특히 팬덤 간에는 본인 팬덤의 전력을 분석하거나 상대 팬덤의 활동을 예측하기 위해 치열한 정보전이 벌어지기도 합니다.  이 과정에서, 팬덤들에게는 인증된 회원만 접근 가능한 팬 사이트가 중요한 역할을 합니다. 이러한 사이트는 음악방송 점수 예측에 필요한 데이터를 취합하여 팬덤에게 정보를 제공하고, 이를 바탕으로 투표 전략 등을 논의할 수 있는 공간을 제공합니다.  따라서 본 프로젝트에서는 음악 아티스트 팬덤의 입장에서, 음악방송 순위에 영향을 주는 데이터를 수집·취합하고, 회원들에게만 데이터를 제공하는 팬 사이트를 위한 데이터베이스를 설계하였습니다.  **2. 개념적 설계 및 논리적 설계**  개념적 설계 단계에서는 ER 다이어그램을 작성하였고, 이후 논리적 설계 단계에서 이를 기반으로 릴레이션 스키마를 정의하였습니다. 이 과정에서 현재 설계한 데이터베이스가 제2정규형(2NF)과 제3정규형(3NF)을 충분히 만족하지 않는 부분이 있다는 것을 발견하였습니다.  기존 설계에서는 스트리밍 데이터에 가수, 곡 제목, 음반 이름 등의 정보가 포함되어 있었습니다. 그러나 이러한 설계는 데이터 중복과 일관성 문제를 초래할 가능성이 높았습니다. 따라서 정규화를 적용하여, 곡 정보와 음반 정보를 별도의 테이블로 분리하였습니다. 이를 통해 참조 무결성을 유지하고, 데이터의 일관성을 보장할 수 있었습니다.  또한, 앱 투표 현황 릴레이션의 경우 초기에는 모든 투표 앱의 데이터를 포함하려 했으나, 구조가 지나치게 복잡해질 가능성을 고려하여 설계를 단순화하였습니다. 현재는 특정 앱의 투표 데이터를 관리하는 데 중점을 둔 구조로 변경하였습니다.  이러한 정규화 작업과 단순화를 통해 데이터베이스의 효율성과 유지보수성을 높이는 방향으로 설계를 개선하였습니다.  **3. 쿼리 생성 및 데이터 입력**  릴레이션 스키마에 따라 각 테이블을 생성한 후, 튜플(행) 데이터를 채우기 위해 데이터를 준비하였습니다.  가장 먼저, Song 릴레이션이 Album 릴레이션의 AlbumID를 참조하고, 다른 릴레이션들이 SongID를 참조하는 구조이므로, 참조 무결성을 유지하기 위해 Album 테이블부터 작성하였습니다.  Album 및 Song 테이블의 데이터는 실제 음악 데이터를 참조하여 생성하였으며, 추가적으로 가상의 아티스트인 \*\*'별빛소녀'\*\*를 추가하여 이 사이트가 별빛소녀의 팬클럽을 위한 사이트라는 컨셉에 맞도록 설정하였습니다. 이를 통해 팬덤 중심의 데이터베이스 설계 목적을 강조할 수 있도록 구성하였습니다.  회원 및 게시글 관련 데이터는 직접 데이터를 작성해보는 방식과 인공지능을 활용해 생성하는 방식을 병행하여 작성하였습니다. 인공지능을 활용해 데이터를 빠르게 생성할 수 있었지만, 세부적인 조정이 필요한 경우에는 직접 작성하는 방식이 더 적합하였습니다.  4. 통계쿼리 작성  팬덤 입장에서 실질적으로 도움이 되는 정보들을 위주로 통계 쿼리를 작성하였습니다.  1) 음악방송 총점수 계산  2) 스트리밍 추이  3) 투표 증가수 분석  4) 특정 기간 활동순위  5) 특정 사용자의 사이트 이용목록  이들 중 가장 중요한 음악방송 총점수는 스트리밍/음반/유튜브/투표 점수가 각각 50/20/20/10 비중으로 반영되어 합산됩니다.  또한 각 점수는 비율제로, 1위가 만점이고 다른 아티스트는 1위에 대한 비율로 점수가 주어집니다. 만약 투표점수가 A그룹 1000표, B그룹 50표라면 총점에 10점/0.5점씩 반영됩니다. |
| **개**  **념**  **적**  **설**  **계** | - 주제에 적합한 ER다이어그램 포함   1. **요구 사항 분석 - [음악방송 점수 현황 팬클럽 사이트 데이터베이스]**   실시간 차트와 음악방송 점수 현황을 공유하고 음악방송 투표 전략과 기타 소통 등을 하는 사이트를 위한 데이터베이스를 만든다.  **1) 데이터 수집 및 관리**  - 스트리밍 데이터는 곡 번호와 날짜로 식별하며, 스트리밍 수를 저장해야 한다.  - 음반 판매량은 음반 번호와 날짜로 식별하며, 판매량 수를 저장해야 한다.  - 유튜브 조회수는 곡 번호와 날짜로 식별하며, 조회수를 저장해야 한다.  - 앱 투표 현황은 곡 번호와 날짜로 식별하며, 투표수를 저장해야 한다.  **2) 곡 및 음반 관리**  - 곡 정보는 곡 번호로 식별하며, 곡 번호, 가수, 곡 제목, 음반 번호를 관리해야 한다.  - 음반 정보는 음반 번호로 식별하며, 음반 번호와 음반 이름을 관리해야 한다.  **3) 사용자 관리**  - 사용자는 회원 가입 시 회원아이디, 비밀번호, 닉네임, 인증자료를 입력해야 한다.  - 가입신청자는 초기 등급이 waiting이며, 관리자가 등급을 member로 변경해야 회원이 된다.  - 관리자의 등급은 admin이어야 한다.  **4) 게시글 및 댓글 관리**  - 회원은 게시글을 여러 개 작성할 수 있으며, 하나의 게시글은 한 명의 회원만 작성할 수 있어야 한다.  - 회원은 댓글을 여러 개 작성할 수 있으며, 하나의 댓글은 한 명의 회원만 작성할 수 있어야 한다.  - 게시글 하나에는 여러 개의 댓글이 작성될 수 있으며, 하나의 댓글은 하나의 게시글에만 작성될 수 있다.  - 게시글은 글 번호로 식별하며, 글 번호, 글 제목, 내용, 작성자, 작성일자, 삭제 여부를 관리해야 한다.  - 댓글은 댓글 번호로 식별하며, 댓글 번호, 글 번호, 내용, 작성자, 작성일자, 삭제 여부를 관리해야 한다.   1. **개념적 설계 - [ER 다이어그램]** |
| **논**  **리**  **적**  **설**  **계** | - 릴레이션 스키마   1. **논리적 설계 - [릴레이션 스키마]** 2. 곡 정보 (Song)  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 테이블 이름 | | | **Song** | | | | | 속성 이름 | 데이터 타입 | 널 허용 여부 | 기본값 | 기본키 | 외래키 | 제약조건 | | SongID | INT | NOT NULL |  | PK |  |  | | Artist | VARCHAR(50) | NOT NULL |  |  |  |  | | SongTitle | VARCHAR(50) | NOT NULL |  |  |  |  | | AlbumID | INT | YES | NULL |  | Album (AlbumID) |  |  1. 음반 정보 (Album)  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 테이블 이름 | | | **Album** | | | | | 속성 이름 | 데이터 타입 | 널 허용 여부 | 기본값 | 기본키 | 외래키 | 제약조건 | | AlbumID | INT | NOT NULL |  | PK |  |  | | AlbumTitle | VARCHAR(50) | NOT NULL |  |  |  |  |  1. 스트리밍 데이터 (StreamingData)  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 테이블 이름 | | | **StreamingData** | | | | | 속성 이름 | 데이터 타입 | 널 허용 여부 | 기본값 | 기본키 | 외래키 | 제약조건 | | SongID | INT | NOT NULL |  | PART OF PK | Song (SongID) |  | | Date | DATE | NOT NULL |  | PART OF PK |  |  | | StreamingCount | INT | NOT NULL | 0 |  |  | 0 이상 |  1. 음반 판매량 (AlbumSales)  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 테이블 이름 | | | **AlbumSales** | | | | | 속성 이름 | 데이터 타입 | 널 허용 여부 | 기본값 | 기본키 | 외래키 | 제약조건 | | AlbumID | INT | NOT NULL |  | PART OF PK | Album (AlbumID) |  | | Date | DATE | NOT NULL |  | PART OF PK |  |  | | SalesCount | INT | NOT NULL | 0 |  |  | 0 이상 |  1. 유튜브 조회수 (YouTubeViews)  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 테이블 이름 | | | **YouTubeViews** | | | | | 속성 이름 | 데이터 타입 | 널 허용 여부 | 기본값 | 기본키 | 외래키 | 제약조건 | | SongID | INT | NOT NULL |  | PART OF PK | Song (SongID) |  | | Date | DATE | NOT NULL |  | PART OF PK |  |  | | ViewCount | INT | NOT NULL | 0 |  |  | 0 이상 |  1. 앱 투표 현황 (AppVotes)  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 테이블 이름 | | | **AppVotes** | | | | | 속성 이름 | 데이터 타입 | 널 허용 여부 | 기본값 | 기본키 | 외래키 | 제약조건 | | SongID | INT | NOT NULL |  | PART OF PK | Song (SongID) |  | | Date | DATE | NOT NULL |  | PART OF PK |  |  | | VoteCount | INT | NOT NULL | 0 |  |  | 0 이상 |  1. 회원 (Member) - VerificationData는 실제로는 IMAGE이지만 구현시에는 VARCHAR로 가정  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 테이블 이름 | | | **Member** | | | | | 속성 이름 | 데이터 타입 | 널 허용 여부 | 기본값 | 기본키 | 외래키 | 제약조건 | | MemberID | VARCHAR(20) | NOT NULL |  | PK |  |  | | Password | VARCHAR(255) | NOT NULL |  |  |  |  | | Nickname | VARCHAR(20) | NOT NULL | 익명 |  |  |  | | MemberLevel | VARCHAR(10) | NOT NULL | waiting |  |  | waiting, member, admin | | VerificationData | IMAGE | NOT NULL |  |  |  |  |  1. 게시글 (Post) - Content는 실제로는 멀티미디어가 가능하지만 TEXT로 가정  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 테이블 이름 | | | **Post** | | | | | 속성 이름 | 데이터 타입 | 널 허용 여부 | 기본값 | 기본키 | 외래키 | 제약조건 | | PostID | INT | NOT NULL | AUTO\_  INCREMENT | PK |  |  | | PostTitle | VARCHAR(50) | NOT NULL |  |  |  |  | | Content | TEXT | NOT NULL |  |  |  |  | | AuthorID | VARCHAR(20) | NOT NULL |  |  | Member (MemberID) |  | | PostDate | DATETIME | NOT NULL | CURRENT\_  TIMESTAMP |  |  |  | | Deleted | BOOL | NOT NULL | 0 |  |  |  |  1. 댓글 (Comment)  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 테이블 이름 | | | **Comment** | | | | | 속성 이름 | 데이터 타입 | 널 허용 여부 | 기본값 | 기본키 | 외래키 | 제약조건 | | CommentID | INT | NOT NULL | AUTO\_  INCREMENT | PK |  |  | | PostID | INT | NOT NULL |  |  | Post (PostID) |  | | Content | TEXT | NOT NULL |  |  |  |  | | AuthorID | VARCHAR(20) | NOT NULL |  |  | Member (MemberID) |  | | PostDate | DATETIME | NOT NULL | CURRENT\_  TIMESTAMP |  |  |  | | Deleted | BOOL | NOT NULL | 0 |  |  |  | |
| **구**  **현**  **쿼**  **리** | - CREATE TABLE              -INSERT SQL |
| **통**  **계**  **쿼**  **리** | -SELECT SQL (JOIN, 다양한 집계 함수 활용)  **1) 음악방송 총점수 계산**        **2) 스트리밍 추이**    **3) 투표 증가수 분석**    **4) 특정 기간 활동순위**    **5) 특정 사용자의 사이트 이용목록** |
| **쿼**  **리**  **실**  **행**  **결**  **과**  **이**  **미**  **지** | **1) 음악방송 총점수 계산**    **2) 스트리밍 추이**    **3) 투표 증가수 분석**    **4) 특정 기간 활동순위**    **5) 특정 사용자의 사이트 이용목록** |
| **기**  **타** | - 항목에 포함되지 않지만 추가로 언급하고 싶은 내용을 작성하세요 |