사킬라 샘플 데이터베이스

목차

이 문서에서는 사킬라 샘플 데이터베이스 설치, 구조, 사용 및 기록에 대해 설명합니다. 법적 정보는 법적 고지 사항을 참조하세요.

MySQL 사용에 대한 도움이 필요하면 다른 MySQL 사용자와 문제를 논의할 수 있는 MySQL 포럼을 방문하세요.

문서 생성 날짜: 2024-12-24(개정: 80591)

1 서문 및 법적 고지

이 문서에서는 사킬라 샘플 데이터베이스 설치, 구조, 사용법 및 기록에 대해 설명합니다.

법적 고지

여기에 포함된 정보는 사전 통지 없이 변경될 수 있으며 오류가 없음을 보증하지 않습니다. 오류를 발견하면 서면으로 알려주시기 바랍니다.

제한된 권한 고지

미국 정부 또는 미국 정부를 대신하여 라이선스를 부여하는 자에게 제공되는 소프트웨어, 소프트웨어 문서, 데이터(연방정부 획득 규정에 정의된 대로) 또는 관련 문서인 경우 다음 고지가 적용됩니다:

미국 정부 최종 사용자: 오라클 프로그램(모든 운영 체제, 통합 소프트웨어, 제공된 하드웨어에 내장, 설치 또는 활성화된 모든 프로그램 및 그러한 프로그램의 수정 사항 포함) 및 오라클 컴퓨터 설명서 또는 미국 정부 최종 사용자가 제공하거나 액세스하는 기타 오라클 데이터는 해당 연방정부 획득 규정에 따라 "상용 컴퓨터 소프트웨어", "상용 컴퓨터 소프트웨어 설명서" 또는 "제한적 권리 데이터"로 간주되며, 다음과 같이 정의됩니다.
기관별 보충 규정. 따라서 i) 오라클 프로그램(모든 운영 체제, 통합 소프트웨어, 제공된 하드웨어에 내장, 설치 또는 활성화된 모든 프로그램 및 그러한 프로그램의 수정 포함), ii) 오라클 컴퓨터 문서 및/또는 iii) 기타 오라클 데이터의 사용, 복제, 배포, 전시, 공개, 수정, 파생 저작물의 준비 및/또는 개조는 해당 계약에 포함된 라이선스에 지정된 권리 및 제한의 적용을 받습니다.
미국 정부의 오라클 클라우드 서비스 사용에 적용되는 약관은 해당 서비스에 대한 해당 계약에 정의되어 있습니다. 미국 정부에는 다른 어떠한 권리도 부여되지 않습니다.

위험 애플리케이션 고지

이 소프트웨어 또는 하드웨어는 다양한 정보 관리 애플리케이션에서 일반적으로 사용하도록 개발되었습니다. 개인 상해의 위험이 있는 애플리케이션을 포함하여 본질적으로 위험한 애플리케이션에서 사용하도록 개발되거나 의도되지 않았습니다. 위험한 애플리케이션에서 이 소프트웨어 또는 하드웨어를 사용하는 경우, 귀하는 안전한 사용을 보장하기 위해 모든 적절한 장애 안전, 백업, 이중화 및 기타 조치를 취할 책임이 있습니다. 오라클과 그 계열사는 위험한 애플리케이션에서 본 소프트웨어 또는 하드웨어를 사용함으로써 발생하는 손해에 대해 어떠한 책임도 지지 않습니다.

상표 고지

Oracle, Java, MySQL 및 NetSuite는 Oracle 및/또는 그 계열사의 등록 상표입니다. 기타 명칭은 해당 소유자의 상표일 수 있습니다.

인텔 및 인텔 인사이드는 인텔 코퍼레이션의 상표 또는 등록 상표입니다. 모든 SPARC 상표는 라이선스 하에 사용되며 SPARC International, Inc.의 상표 또는 등록 상표입니다. AMD, Epyc 및 AMD 로고는 Advanced Micro Devices의 상표 또는 등록 상표입니다. UNIX는 The Open Group의 등록 상표입니다.

타사 콘텐츠, 제품 및 서비스 면책 조항

본 소프트웨어 또는 하드웨어 및 설명서는 제3자의 콘텐츠, 제품 및 서비스에 대한 액세스 또는 정보를 제공할 수 있습니다. 오라클과 그 계열사는 귀하와 오라클 간의 해당 계약에 달리 명시되지 않는 한 타사 콘텐츠, 제품 및 서비스와 관련된 모든 종류의 보증에 대해 책임을 지지 않으며 명시적으로 보증을 부인합니다. 오라클과 그 계열사는 귀하와 오라클 간의 해당 계약에 명시된 경우를 제외하고 귀하가 타사 콘텐츠, 제품 또는 서비스에 액세스하거나 사용함으로 인해 발생하는 손실, 비용 또는 손해에 대해 책임을 지지 않습니다.

이 문서의 사용

이 문서는 GPL 라이선스에 따라 배포되지 않습니다. 이 문서의 사용에는 다음 약관이 적용됩니다:

이 설명서의 인쇄본은 개인적인 용도로만 사용할 수 있습니다. 실제 콘텐츠를 어떤 식으로든 변경하거나 편집하지 않는 한 다른 형식으로 변환하는 것은 허용됩니다. 다음 사항을 게시해서는 안 됩니다.

접근성을 위한 오라클 지원 액세스

지원을 구매한 오라클 고객은 나의 오라클 지원을 통해 전자 지원을 이용할 수 있습니다. 자세한 내용은 다음을 방문하십시오. http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info 또는 http://www.oracle.com/pls/topic/ lookup?ctx=acc&id=trs(청각 장애가 있는 경우)를 방문하세요.

2 소개

사킬라 샘플 데이터베이스는 MySQL AB 문서화 팀의 전 멤버였던 마이크 힐러가 처음에 개발했습니다. 이 데이터베이스는 책, 튜토리얼, 기사, 샘플 등의 예제에 사용할 수 있는 표준 스키마를 제공하기 위한 것입니다. 또한 Sakila 샘플 데이터베이스는 보기, 저장 프로시저 및 트리거와 같은 MySQL의 기능을 강조하는 역할을 합니다.

사킬라 샘플 데이터베이스와 그 사용법에 대한 추가 정보는 MySQL 포럼을 통해 확인할 수 있습니다.

사킬라 샘플 데이터베이스는 MySQL 사용자 커뮤니티의 지원과 피드백의 결과이며, 피드백 및 사용자 의견은 언제나 환영합니다. 모든 피드백은 http://www.mysql.com/ company/contact/로 보내주세요. 버그 보고는 MySQL 버그를 이용하세요.

3 역사

사킬라 샘플 데이터베이스는 오라클에서 제공하는 월드 샘플 데이터베이스를 대체하기 위해 설계되었습니다.

세계 샘플 데이터베이스는 전 세계 국가 및 도시에 대한 정보가 포함된 테이블 집합을 제공하며 기본 쿼리에는 유용하지만 MySQL 5 이상에 있는 MySQL 관련 기능 및 기능을 테스트할 수 있는 구조가 부족합니다.

사킬라 샘플 데이터베이스의 개발은 2005년 초에 시작되었습니다. 초기 설계는 Dell Dell PowerEdge MySQL 애플리케이션에 대한 세 가지 접근 방식에서 백서인 서버의 사용된 데이터베이스를 기반으로 했습니다.

Dell의 샘플 데이터베이스가 온라인 DVD 스토어를 대표하도록 설계되었다면, Sakila 샘플 데이터베이스는 DVD 대여점을 대표하도록 설계되었습니다. 사킬라 샘플 데이터베이스는 여전히 Dell 샘플 데이터베이스에서 영화 및 배우 이름을 차용합니다.

개발은 스키마 설계를 위해 MySQL 쿼리 브라우저를 사용하여 이루어졌으며, 테이블은 기여자의 노력과 더불어 MySQL 쿼리 브라우저와 사용자 정의 스크립트의 조합으로 채워졌습니다(섹션 8, '감사의 말' 참조).

기본 스키마가 완성된 후, 다양한 뷰, 저장 루틴, 트리거가 스키마에 추가되고 샘플 데이터가 채워졌습니다. 일련의 검토 버전을 거친 후, 2006년 3월에 Sakila 샘플 데이터베이스의 첫 번째 공식 버전이 출시되었습니다.

4 설치

사킬라 샘플 데이터베이스는 https://dev.mysql.com/doc/index-other.html 에서 확인할 수 있습니다. 다운로드 가능한 아카이브는 압축된 `tar` 파일 또는 Zip 형식으로 제공됩니다. 아카이브에는 `sakila- schema.sql`, `sakila-data.sql`, `sakila.mwb`의 세 가지 파일이 포함되어 있습니다.

> **참고**
>
> 사킬라에는 MySQL 버전별 주석이 포함되어 있는데, 이는 사킬라 스키마 및 데이터가 MySQL 서버의 버전에 따라 달라지기 때문입니다. 예를 들어, MySQL 서버 5.7.5는 `InnoDB`에 공간 데이터 인덱싱을 지원하므로 **주소** 테이블에 MySQL 5.7.5 이상에 대한 공간 인식 **위치** 열이 포함됩니다.

`sakila-schema.sql` 파일에는 테이블, 보기, 저장 프로시저 및 트리거를 포함하여 Sakila 데이터베이스의 구조를 만드는 데 필요한 모든 `CREATE` 문이 포함되어 있습니다.

`sakila-data.sql` 파일에는 초기 데이터 로드 후에 생성해야 하는 트리거에 대한 정의와 함께 `sakila-schema.sql` 파일에서 생성한 구조를 채우는 데 필요한 `INSERT` 문이 포함되어 있습니다.

`sakila.mwb` 파일은 MySQL Workbench 내에서 열어 데이터베이스 구조를 검사할 수 있는 MySQL Workbench 데이터 모델입니다. 자세한 내용은 MySQL Workbench를 참조하세요.

Sakila 샘플 데이터베이스를 설치하려면 다음 단계를 따르세요:

1. 설치 아카이브를 `C:\temp\` 또는 `/tmp/`와 같은 임시 위치에 압축을 풉니다. 아카이브의 압축을 풀면 `sakila-db`라는 디렉터리가 생성되며, 이 `sakila-schema.sql` 및 `sakila-data.sql` 파일이 들어 있습니다.

2. 다음 명령어를 사용하여 `mysql` 명령줄 클라이언트를 사용하여 MySQL 서버에 연결합니다:

   ```
   $> mysql -u root -p
   ```

   메시지가 표시되면 비밀번호를 입력합니다. 새 데이터베이스를 만들 수 있는 권한이 있는 경우 루트 계정이 아닌 계정도 사용할 수 있습니다.

3. 다음 명령을 사용하여 `sakila-schema.sql` 스크립트를 실행하여 데이터베이스 구조를 생성하고, `sakila- data.sql` 스크립트를 실행하여 데이터베이스 구조를 채웁니다:

   ```
   mysql> SOURCE C:/temp/sakila-db/sakila-schema.sql;
   mysql> SOURCE C:/temp/sakila-db/sakila-data.sql;
   ```

   `sakila-schema.sql` 및 `sakila-data.sql` 파일의 경로를 시스템의 실제 경로로 바꿉니다.

   > **참고**
   >
   > Windows에서는 소스 실행 시 백슬래시 대신 슬래시를 사용합니다.
   > 명령어를 사용합니다.

4. 샘플 데이터베이스가 올바르게 설치되었는지 확인합니다. 다음 문을 실행합니다. 여기에 표시된 것과 유사한 출력이 표시되어야 합니다.

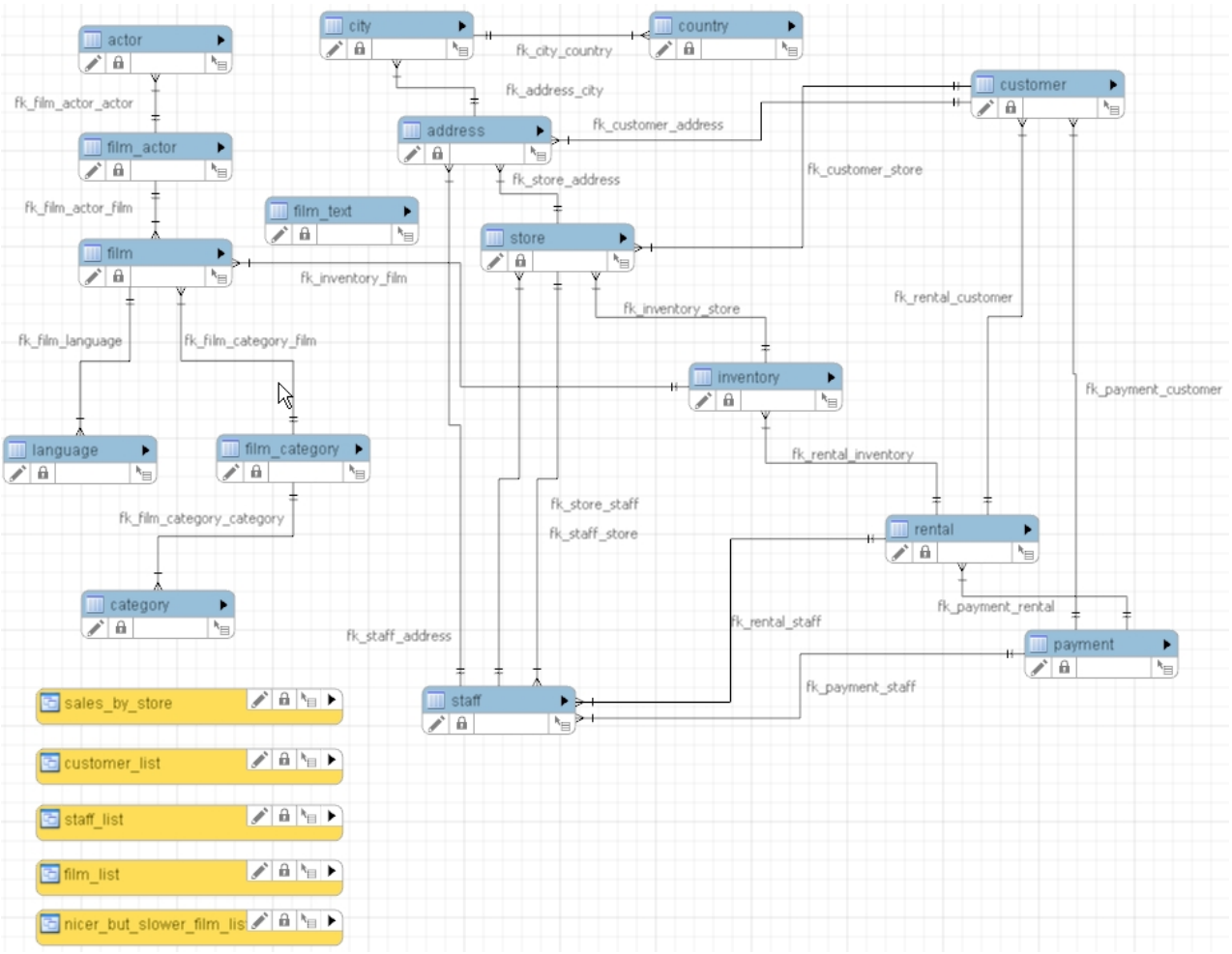
   ```
   mysql> USE sakila;
   데이터베이스 변경

   mysql> 전체 테이블 표시;
   ```

```
+----------------------------+----------------------------+
| Tables_in_sakila           | 테이블 유형 |
+----------------------------+----------------------------+
| 배우                       | 기본 테이블 |
| 배우_정보                  | 보기 |
| 주소                       | 기본 테이블 |
| 카테고리                   | 기본 테이블 |
| 도시                       | 기본 테이블 |
| 국가                       | 기본 테이블 |
| 고객                       | 기본 테이블 |
| customer_list              | 기본 테이블 |
| 영화                       | 기본 테이블 |
| film_actor                 | 기본 테이블 |
| 영화_카테고리              | 기본 테이블 |
| film_list                  | 기본 테이블 |
| film_text                  | 기본 테이블 |
| 인벤토리                   | 기본 테이블 |
| 더_좋은_하지만_느린_파일_목록 | 보기 |
| 결제                       | 기본 테이블 |
| 렌탈                       | 기본 테이블 |
| 판매_별_필름_카테고리      | 보기 |
| 판매_별_매장               | 보기 |
| 직원                       | 기본 테이블 |
| 직원_목록                  | 보기 |
| store                      | 기본 테이블 |
+----------------------------+----------------------------+
한 세트에 23행(0.01초)

mysql> SELECT COUNT(*) FROM film;
+ -------------+
| COUNT(*) |
+ -------------+
|       |
+ -------------+
한 세트에 1행(0.00초)

mysql> SELECT COUNT(*) FROM film_text;
+ -------------+
| COUNT(*) |
+ -------------+
|       |
+ -------------+
한 세트에 1행(0.00초)
```

5 구조

다음 다이어그램은 Sakila 샘플 데이터베이스 구조에 대한 개요를 제공합니다. 다이어그램 소스 파일(MySQL Workbench와 함께 사용하기 위한)은 Sakila 배포에 포함되어 있으며 이름은 `sakila.mwb`입니다.

그림 1 사킬라 스키마



5.1 테이블

다음 섹션에서는 사킬라 샘플 데이터베이스를 구성하는 테이블을 알파벳 순서대로 설명합니다.

5.1.1 액터 테이블

액터 테이블에는 모든 액터에 대한 정보가 나열됩니다.

배우 `film_actor` 영화 테이블을 통해 테이블에 조인됩니다.

열

- `ACTOR_ID`: 테이블의 각 액터를 고유하게 식별하는 데 사용되는 대리 기본 키입니다.
- `first_name`: 배우의 이름입니다.
- `last_name`: 배우의 성입니다.
- `last_update`: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.1.2 주소 테이블

주소 테이블에는 고객, 직원 및 스토어에 대한 주소 정보가 포함되어 있습니다.

주소 테이블 기본 키는 고객, 직원 및 스토어 테이블에서 외래 키로 나타납니다.

열

- address_id: 테이블의 각 주소를 고유하게 식별하는 데 사용되는 대리 기본 키입니다.
- 주소: 주소의 첫 번째 줄입니다.
- 주소2: 주소의 두 번째 줄(선택 사항)입니다.
- 지구: 주소의 지역으로, 주, 도, 현 등이 될 수 있습니다.
- city_id: 도시 테이블을 가리키는 외래 키입니다.
- 우편번호: 주소의 우편번호 또는 우편번호(해당되는 경우).
- 전화: 주소의 전화번호입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.
- 위치: 공간 인덱스가 있는 지오메트리 열입니다.

> **참고**
>
> 공간 위치 열은 MySQL 5.7.5부터 지원됩니다. 이 열은 MySQL 서버 5.7.5 이상에서 Sakila SQL 파일을 실행할 때만 추가됩니다. 또한 공간 키 idx_location도 추가됩니다.

5.1.3 카테고리 표

카테고리 표에는 동영상에 할당할 수 있는 카테고리가 나열되어 있습니다.

카테고리 테이블은 film_category 영화 테이블을 통해 테이블에 조인됩니다.

열

- category_id: 테이블의 각 카테고리를 고유하게 식별하는 데 사용되는 대리 기본 키입니다.
- 이름: 카테고리의 이름입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.1.4 도시 테이블

도시 표에는 도시 목록이 포함되어 있습니다.

도시 테이블은 주소 테이블에서 외래 키로 참조되며 외래 키를 사용하여 국가 테이블을 참조합니다.

열

- city_id: 테이블의 각 도시를 고유하게 식별하는 데 사용되는 대리 기본 키입니다.
- 도시: 도시 이름입니다.
- country_id: 도시가 속한 국가를 식별하는 외래 키입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시기입니다.

5.1.5 국가 표

국가 표에는 국가 목록이 포함되어 있습니다.

국가 테이블은 도시 테이블의 외래 키로 참조됩니다.

열

- country_id: 테이블의 각 국가를 고유하게 식별하는 데 사용되는 대리 기본 키입니다.
- 국가: 국가 이름입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.1.6 고객 테이블

고객 테이블에는 모든 고객 목록이 포함되어 있습니다.

고객 테이블은 결제 및 렌탈 테이블에서 참조되며 주소 및
외래 키를 사용하여 테이블을 저장합니다.

열

- customer_id: 테이블에서 각 고객을 고유하게 식별하는 데 사용되는 대리 기본 키입니다.
- store_id: 고객 "홈 스토어"를 식별하는 외래 키입니다. 고객이 이 스토어에서만 렌탈하는 것은 아니지만 일반적으로 쇼핑하는 스토어입니다.
- first_name: 고객 이름입니다.
- last_name: 고객 성입니다.
- 이메일: 고객 이메일 주소입니다.
- address_id: 주소 테이블에서 고객 주소를 식별하는 외래 키입니다.
- 활성: 고객이 활성 고객인지 여부를 나타냅니다. 이 값을 FALSE로 설정하면 고객을 완전히 삭제하는 대신 사용할 수 있습니다. 대부분의 쿼리에는 WHERE 활성= TRUE 절이 있어야 합니다.
- create_date: 고객이 시스템에 추가된 날짜입니다. 이 날짜는 INSERT 중 트리거를 사용하여 자동으로 설정됩니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시기입니다.

5.1.7 영화 테이블

필름 표는 스토어에 잠재적으로 재고가 있는 모든 필름의 목록입니다. 각 필름의 실제 재고 사본은 재고 표에 표시됩니다.

영화 테이블은 언어 테이블을 참조하며 film_category, film_actor 및 인벤토리 테이블에서 참조합니다.

열

- film_id: 테이블의 각 필름을 고유하게 식별하는 데 사용되는 대리 기본 키입니다.
- 제목: 영화의 제목입니다.
- 설명: 설명: 영화에 대한 간단한 설명 또는 줄거리 요약입니다.

- release_year: 영화가 개봉된 연도입니다.
- language_id: 언어 테이블을 가리키는 외래 키로, 동영상의 언어를 식별합니다.
- original_language_id: 언어 테이블을 가리키는 외래 키로, 영화의 원래 언어를 식별합니다. 영화가 새로운 언어로 더빙되었을 때 사용됩니다.
- rental_duration: 대여 기간(일 단위)입니다.
- rental_rate: 대여 기간 열에 지정된 기간 동안 영화를 대여하는 데 드는 비용입니다.
- 길이: 필름의 길이(분)입니다.
- 교체_비용: 반환되지 않거나 손상된 상태로 반환되는 경우 고객에게 청구되는 금액입니다.
- 등급: 등급: 동영상에 부여된 등급입니다. 다음 중 하나가 될 수 있습니다: G, PG, PG-13, R 또는 NC-17.
- special_features: DVD에 포함된 일반적인 특수 기능을 나열합니다. 0개 이상일 수 있습니다: 예고편, 코멘터리, 삭제 장면, 비하인드 스토리.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.1.8 film_actor 테이블

film_actor 테이블은 영화와 배우 간의 다대다 관계를 지원하는 데 사용됩니다. 특정 영화의 각 배우에 대해 film_actor 테이블에는 배우와 영화가 나열된 행이 하나씩 있습니다.

film_actor 테이블은 외래 키를 사용하여 영화 및 배우 테이블을 참조합니다.

열:

- actor_id: 액터를 식별하는 외래 키입니다.
- film_id: 필름을 식별하는 외래 키입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.1.9 film_category 테이블

film_category 테이블은 영화와 카테고리 간의 다대다 관계를 지원하는 데 사용됩니다. 영화에 적용된 각 카테고리에 대해 film_category 테이블에는 카테고리와 영화가 나열된 행이 하나씩 있습니다.

film_category 테이블은 외래 키를 사용하는 영화 및 카테고리 테이블을 참조합니다.

열:

- film_id: 필름을 식별하는 외래 키입니다.
- category_id: 카테고리를 식별하는 외래 키입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.1.10 film_text 테이블

film_text 테이블에는 영화 테이블의 film_id, 제목 및 설명 열이 포함되며, 영화 테이블 INSERT, UPDATE 및 DELETE 작업의 트리거를

통해 테이블의 내용이 영화 테이블과 동기화 상태로 유지됩니다(5.5절. "트리거" 참조).

MySQL 서버 5.6.10 이전에는 영화_text 테이블이 Sakila 샘플 데이터베이스에서 MyISAM 스토리지 엔진을 사용하는 유일한 테이블이었습니다. 이는 영화 테이블에 나열된 영화의 제목과 설명에 전체 텍스트 검색이 사용되기 때문입니다. MySQL 서버 5.6.10까지는 InnoDB를 통한 전체 텍스트 검색 지원이 제공되지 않았기 때문에 MyISAM이 사용되었습니다.

열

- film_id: 테이블의 각 필름을 고유하게 식별하는 데 사용되는 대리 기본 키입니다.
- 제목: 영화의 제목입니다.
- 설명: 설명: 영화에 대한 간단한 설명 또는 줄거리 요약입니다.

film_text 테이블의 콘텐츠는 직접 수정해서는 안 됩니다. 모든 변경은 필름 테이블에서 수행해야 합니다.

5.1.11 인벤토리 테이블

재고 테이블에는 지정된 스토어에서 지정된 필름의 각 사본에 대해 한 행이 포함됩니다.

인벤토리 테이블은 외래 키를 사용하는 필름 및 저장소 테이블을 참조하며
대여 테이블.

열

- inventory_id: 인벤토리의 각 아이템을 고유하게 식별하는 데 사용되는 대리 기본 키입니다.
- film_id: 이 항목이 나타내는 필름을 가리키는 외래 키입니다.
- store_id: 이 아이템을 보유한 스토어를 가리키는 외래 키입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.1.12 언어 표

언어 표는 영화의 언어 및 원어 값에 대해 가능한 언어가 나열된 조회 테이블입니다.

언어 표는 영화 표에서 참조합니다.

열

- language_id: 각 언어를 고유하게 식별하는 데 사용되는 대리 기본 키입니다.
- 이름: 언어의 영문 이름입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.1.13 결제 테이블

결제 테이블에는 고객이 결제한 각 결제 내역이 금액 및 결제 중인 대여료(해당되는 경우) 등의 정보와 함께 기록됩니다.

결제 테이블은 고객, 대여 및 직원 테이블을 의미합니다.

열

- payment_id: 각 결제를 고유하게 식별하는 데 사용되는 대리 기본 키입니다.

- customer_id: 결제가 잔액이 있는 고객입니다. 고객 테이블에 대한 외래 키 참조입니다.
- staff_id: 결제를 처리한 직원입니다. 직원에 대한 외래 키 참조입니다.
  테이블.
- rental_id: 결제가 적용되고 렌탈입니다. 일부 결제는 미결제 요금에 대한 결제이며 대여와 직접 관련이 없을 수 있으므로 선택 사항입니다.
- 금액: 금액: 결제 금액입니다.
- 결제_날짜: 결제가 처리된 날짜입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.1.14 대여 테이블

대여 테이블에는 누가 어떤 품목을 대여했는지, 언제 대여했는지, 언제 반납했는지에 대한 정보가 포함된 각 인벤토리 품목의 대여마다 한 행이 포함됩니다.

렌탈 테이블은 재고, 고객 및 직원 테이블을 의미하며 다음과 같이 참조됩니다.
결제 테이블.

열

- rental_id: 렌탈을 고유하게 식별하는 대리 기본 키입니다.
- 대여_날짜: 아이템을 대여한 날짜와 시간입니다.
- 인벤토리_ID: 대여 중인 아이템입니다.
- customer_id: 아이템을 대여하는 고객입니다.
- return_date: 아이템이 반품된 날짜와 시간입니다.
- staff_id: 대여를 처리한 직원입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.1.15 직원 테이블

직원 테이블에는 이메일 주소, 로그인 정보, 사진 등 모든 직원 정보가 나열됩니다.

직원 테이블은 외래 키를 사용하는 상점 및 주소 테이블을 참조하며, 외래 키를 사용하는
대여, 결제 및 스토어 테이블

열

- staff_id: 직원을 고유하게 식별하는 대리 기본 키입니다.
- first_name: 직원의 이름입니다.
- last_name: 직원의 성입니다.
- address_id: 주소 테이블에 있는 직원 주소의 외래 키입니다.
- 사진: 직원의 사진이 포함된 BLOB입니다.
- 이메일: 직원 이메일 주소입니다.

- store_id: "홈 스토어" 직원입니다. 직원은 다른 스토어에서 근무할 수 있지만 일반적으로 나열된 스토어에 배정됩니다.
- active: 활성 직원인지 여부입니다. 직원이 퇴사하면 이 테이블에서 해당 행이 삭제되지 않고 대신 이 열이 FALSE로 설정됩니다.
- 사용자 이름: 직원이 대여 시스템에 액세스하는 데 사용하는 사용자 이름입니다.
- 비밀번호: 직원이 대여 시스템에 액세스하는 데 사용하는 비밀번호입니다. 비밀번호는 SHA2() 함수를 사용하여 해시로 저장해야 합니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시기입니다.

5.1.16 스토어 테이블

스토어 테이블에는 시스템의 모든 스토어가 나열됩니다. 모든 재고는 특정 스토어에 할당되며 직원과 고객에게는 "홈 스토어"가 할당됩니다.

스토어 테이블은 외래 키를 사용하는 직원 및 주소 테이블을 참조하고
직원, 고객, 재고 테이블을 만들 수 있습니다.

열

- store_id: 스토어를 고유하게 식별하는 대리 기본 키입니다.
- 매니저_스태프_ID: 이 스토어의 관리자를 식별하는 외래 키입니다.
- address_id: 이 상점의 주소를 식별하는 외래 키입니다.
- last_update: 행이 생성되었거나 가장 최근에 업데이트된 시점입니다.

5.2 조회수

다음 섹션에서는 사킬라 샘플 데이터베이스에 포함된 보기를 알파벳 순서대로 설명합니다.

5.2.1 배우_정보 보기

배우_정보 보기는 배우가 출연했던 영화를 포함한 모든 배우의 목록을 카테고리별로 분류하여 제공합니다.

스태프_목록 뷰에는 영화, 배우, 카테고리, 영화_배우 및
영화_카테고리 테이블.

5.2.2 고객 목록 보기

customer_list 보기는 이름과 성이 연결된 고객 목록과 주소 정보가 하나의 보기로 결합된 고객 목록을 제공합니다.

customer_list 뷰에는 고객, 주소, 도시 및 국가 테이블의 데이터가 통합되어 있습니다.

5.2.3 film_list 보기

film_list 보기에는 각 영화에 대한 배우 목록이 쉼표로 구분된 영화 테이블의 형식이 지정된 보기가 포함되어 있습니다.

film_list 보기에는 영화, 카테고리, 영화_카테고리, 배우, 배우의 데이터가 통합됩니다.
film_actor 테이블.

5.2.4 더 좋은_하지만_느린_파일 목록 보기

`nicer_but_slower_film_list` 뷰에는 쉼표로 구분된 영화 배우 목록이 포함된 영화 테이블의 형식화된 보기가 포함되어 있습니다.

`nicer_but_slower_film_list` 보기는 배우 목록의 `film_list` 보기와 다릅니다. 배우 이름의 대소문자가 모두 대문자로 표시되지 않고 각 이름의 첫 글자가 대문자로 표시되도록 조정됩니다.

이름에서 알 수 있듯이 `nicer_but_slower_film_list` 뷰는 추가 처리를 수행하므로 `film_list` 뷰보다 데이터를 반환하는 데 시간이 더 오래 걸립니다.

더 좋은_하지만_느린_필름 목록 뷰는 영화, 카테고리, 영화_카테고리, 배우 및 영화_배우 테이블의 데이터를 통합합니다.

5.2.5 판매_별_필름_카테고리 뷰

sales_by_film_category 뷰는 개별 영화 카테고리별로 분류된 총 판매 목록을 제공합니다.

하나의 영화가 여러 카테고리에 나열될 수 있으므로 이 뷰의 행을 합산하여 총 매출을 계산하는 것은 바람직하지 않습니다.

판매_별_영화_카테고리 뷰에는 카테고리, 결제, 대여, 재고, 영화, 영화_category 및 카테고리 테이블의 데이터가 통합되어 있습니다.

5.2.6 판매_별_스토어 뷰

`sales_by_store` 뷰는 스토어별로 분류된 총 판매 목록을 제공합니다. 이 뷰는 스토어 위치, 관리자 이름 및 총 판매

액을 반환합니다.

판매_별_스토어 뷰에는 도시, 국가, 결제, 임대, 재고, 스토어, 주소 및 직원 테이블의 데이터가 통합되어 있습니다.

5.2.7 직원 목록 보기

`staff_list` 뷰는 주소 및 스토어 정보를 포함한 모든 목록을 제공합니다. `staff_list` 뷰는 직원 및 주소 테이블의 데이터를 통합합니다.

5.3 저장된 절차

다음 섹션에서는 사킬라 샘플 데이터베이스에 포함된 저장 프로시저를 알파벳 순서대로 설명합니다.

나열된 모든 매개변수는 별도로 나열되지 않는 한 `IN` 매개변수입니다.

5.3.1 필름_인_스톡 저장 프로시저

설명

`film_in_stock` 저장 지정된 스토어에 지정된 영화의 사본이 재고가 있는지 여부를 확인합니다.

매개변수

- `p_film_id`: 필름 테이블의 `film_id` 열에서 확인할 필름의 ID입니다.

- p_store_id: 스토어 테이블의 store_id 열에서 확인할 스토어의 ID입니다.
- p_film_count: 보유 중인 필름 사본의 개수를 반환하는 OUT 매개변수입니다.

반환 값

이 절차는 재고가 있는 필름 사본의 재고 ID 번호 테이블을 생성하고 해당 테이블의 행 수를 나타내는 개수를 반환합니다(p_film_count 매개변수).

샘플 사용법

```
mysql> CALL film_in_stock(1,1,@count);
+--------------------+
| 인벤토리_ID |
+--------------------+
|                  1 |
|                  2 |
|                  3 |
|                  4 |
+--------------------+
한 세트에 4행(0.01초)

쿼리 확인, 영향을 받은 행 1개(0.01초) mysql> SELECT

@count;
+-----------+
| @count |
+-----------+
|         4 |
+-----------+
한 세트에 1행(0.00초)
```

5.3.2 film_not_in_stock 저장 절차

설명

film_not_in_stock 저장 프로시저는 지정된 스토어에 지정된 필름의 재고가 없는(대여된) 사본이 있는지 여부를 확인합니다.

매개변수

- p_film_id: 필름 테이블의 film_id 열에서 확인할 필름의 ID입니다.
- p_store_id: 스토어 테이블의 store_id 열에서 확인할 스토어의 ID입니다.
- p_film_count: 재고가 없는 필름의 사본 수를 반환하는 OUT 매개변수입니다.

반환 값

이 절차는 재고가 없는 필름 사본의 재고 ID 번호 테이블을 생성하고 해당 테이블의 행 수를 나타내는 개수를 반환합니다(p_film_count 매개변수).

샘플 사용법

```
mysql> CALL film_not_in_stock(2,2,@count);
+--------------------+
| 인벤토리_ID |
+--------------------+
|                  9 |
+--------------------+
한 세트에 1행(0.01초)

쿼리 확인, 영향을 받은 행 1개(0.01초)
```

```
mysql> SELECT @count;
+-----------+
| @count |
+-----------+
|         1 |
+-----------+
한 세트에 1행(0.00초)
```

5.3.3 rewards_report 저장 프로시저

설명

`rewards_report` 저장 전월의 상위 고객 목록을 사용자 지정할 수 있는 목록으로 생성합니다.

매개변수

- 최소_월별_구매: 고객이 자격을 얻기 위해 지난 달에 수행해야 했던 최소 구매 또는 대여 횟수입니다.
- 최소_달러_금액_구매: 고객이 자격을 얻기 위해 지난 달에 지출해야 했던 최소 달러 금액입니다.
- 카운트_보상자: 지정된 자격을 충족한 고객 수를 반환하는 `OUT` 매개변수입니다.

반환 값

이 절차는 지정된 자격을 충족하는 고객 테이블을 생성합니다. 이 테이블은 고객 테이블과 동일한 구조를 가집니다. 또한 이 프로시저는 해당 테이블의 행 수를 나타내는 카운트를 카운트_보상자 매개 변수에 반환합니다.

샘플 사용법

```
mysql> CALL rewards_report(7,20.00,@count);
...
| 598              | 1           | WADE                 | 델발           | WADE.DELVALLE@sakilacustomer.org                      |  604
| 599              | 2           | AUSTIN             | CINTRON          | AUSTIN.CINTRON@sakilacustomer.org                     |  605
...


한 세트에 42행(0.11초)

쿼리 완료, 영향을 받은 행 0개(0.67초) mysql> SELECT @count;
+-----------+
| @count |
+-----------+



|        42 |
+-----------+
한 세트에 1행(0.00초)
```

5.4 저장된 함수

다음 섹션에서는 사킬라 샘플 데이터베이스에 포함된 저장 함수에 대해 설명합니다.

5.4.1 get_customer_balance 함수

`get_customer_balance` 함수는 지정된 고객의 계정에서 현재 미결제 금액을 반환합니다.

매개변수

- p_customer_id: 고객의 customer_id 열에서 확인할 고객의 ID입니다.
  테이블.
- p_effective_date: 잔액에 적용될 항목의 마감일입니다. 이 날짜 이후의 대여, 결제 등은 포함되지 않습니다.

반환 값

이 함수는 고객 계정의 미결제 금액을 반환합니다.

샘플 사용법

```
mysql> SELECT get_customer_balance(298,NOW());
+------------------------------------------+
| get_customer_balance(298,NOW()) |
+------------------------------------------+
|                                    22.00 |
+------------------------------------------+
한 세트에 1행(0.00초)
```

5.4.2 inventory_held_by_customer 함수

inventory_held_by_customer 함수는 지정된 인벤토리 품목을 대여한 고객의 customer_id를 반환합니다.

매개변수

- p_inventory_id: 확인할 인벤토리 항목의 ID입니다.

반환 값

이 함수는 현재 대여 중인 고객의 customer_id를 반환하거나, 재고가 있는 경우 NULL을 반환합니다.

샘플 사용법

```
mysql> SELECT inventory_held_by_customer(8);
+ ----------------------------------------+
| 인벤토리_보유_고객별(8) |
+ ----------------------------------------+
|                                   NULL |
+ ----------------------------------------+
한 세트에 1행(0.00초)

mysql> SELECT inventory_held_by_customer(9);
+ ----------------------------------------+
| 인벤토리_보유_고객별(9) |
+ ----------------------------------------+
|                                    366 |
+ ----------------------------------------+
한 세트에 1행(0.00초)
```

5.4.3 인벤토리_인_스톡 함수

인벤토리_함수는 지정한 인벤토리 품목의 재고 여부를 나타내는 부울 값을 반환합니다.

매개변수

- p_inventory_id: 확인할 인벤토리 항목의 ID입니다.

반환 값

이 함수는 지정된 품목의 재고가 있는지 여부를 나타내는 `TRUE` 또는 반환합니다.

샘플 사용법

```
mysql> SELECT inventory_in_stock(9);
+--------------------------------+
| 인벤토리_인_스톡(9) |
+--------------------------------+
|                              0 |
+--------------------------------+
한 세트에 1행(0.00초)

mysql> SELECT inventory_in_stock(8);
+--------------------------------+
| 인벤토리_인_스톡(8) |
+--------------------------------+
|                              1 |
+--------------------------------+
한 세트에 1행(0.00초)
```

5.5 트리거

다음 섹션에서는 사킬라 샘플 데이터베이스의 트리거에 대해 설명합니다.

5.5.1 customer_create_date 트리거

`customer_create_date` 트리거는 행이 삽입될 때 `고객` 테이블의 `create_date` 열을 현재 시간 및 날짜로 설정합니다.

5.5.2 payment_date 트리거

`payment_date` 트리거는 행이 삽입될 때 `결제` 테이블의 `payment_date` 열을 현재 시간 및 날짜로 설정합니다.

5.5.3 rental_date 트리거

`rental_date` 트리거는 행이 삽입될 때 대여 테이블의 `rental_date` 열을 현재 시간 및 날짜로 설정합니다.

5.5.4 ins_film 트리거

`ins_film` 트리거는 `필름` 테이블의 모든 `INSERT` 작업을 `필름_텍스트` 테이블에 복제합니다.

5.5.5 upd_film 트리거

`upd_film` 트리거는 `필름` 테이블의 모든 `업데이트` 작업을 `필름_텍스트` 테이블에 복제합니다.

5.5.6 델_필름 트리거

`del_film` 트리거는 `필름` 테이블의 모든 `DELETE` 작업을 `필름_텍스트` 테이블에 복제합니다.

6 사용 예

다음은 Sakila 샘플 데이터베이스를 사용하여 일반적인 연산을 수행하는 방법에 대한 몇 가지 사용 예시입니다. 이러한 작업은 저장 프로시저 및 뷰에 적합한 후보이지만, 이러한 구현은 의도적으로 사용자의 연습으로 남겨 두었습니다.

- DVD 대여
- DVD 반환
- 기한이 지난 DVD 찾기

DVD 대여

DVD를 대여하려면 먼저 지정된 인벤토리 품목의 재고가 있는지 확인한 다음 대여 테이블에 행을 삽입합니다. 대여 테이블이 생성되면 결제 테이블에 행을 삽입합니다. 비즈니스 규칙에 따라 대여를 처리하기 전에 고객에게 미결제 잔액이 있는지 확인해야 할 수도 있습니다.

```
mysql> SELECT inventory_in_stock(10);
+--------------------------------+
| 인벤토리_인_스톡(10) |
+--------------------------------+
|                              1 |
+--------------------------------+
한 세트에 1행(0.01초)

mysql> INSERT INTO rental(rental_date, inventory_id, customer_id, staff_id) VALUES(NOW(), 10, 3, 1);
쿼리 확인, 영향을 받은 행 1개(0.00초)

mysql> SET @rentID = LAST_INSERT_ID(),
                     @balance = get_customer_balance(3, NOW());
쿼리 확인, 영향을 받은 행 0개(0.14초)

mysql> SELECT @rentID, @balance;
+---------+-----------------+
| @rentID | @balance |
+---------+-----------------+
|    16050|           4.99 |
+---------+-----------------+
한 세트에 1행(0.00초)

mysql> INSERT INTO payment (customer_id, staff_id, rental_id, amount, payment_date) VALUES(3, 1, @rentID, @balance,
               NOW());
쿼리 확인, 영향을 받은 행 1개(0.00초)
```

DVD 반환

DVD를 반납하려면 대여 테이블을 업데이트하고 반납 날짜를 설정하세요. 이렇게 하려면 먼저 반품할 품목의 인벤토리_ID를 기준으로 업데이트할 대여_ID를 식별합니다. 상황에 따라 고객 잔액을 확인하고 결제 테이블에 행을 삽입하여 연체료 결제를 처리해야 할 수도 있습니다.

```
mysql> SELECT rental_id
                      렌탈에서
              WHERE inventory_id= 10 AND
              customer_id = 3
              그리고 반환 날짜가 @rentID에 NULL입니다
              ;
쿼리 확인, 영향을 받은 행 1개(0.01초)

       mysql> SELECT @rentID;
+------------+
| @rentID |
+------------+
|    16050 |
+------------+
```

```
한 세트에 1행(0.00초)

mysql> 임대 업데이트
               SET return_date = NOW() WHERE
               rental_id= @rentID;
쿼리 확인, 영향을 받은 행 1개(0.00초)
행이 일치합니다: 1 변경됨: 1 경고: 0

mysql> SELECT get_customer_balance(3, NOW());
+-----------------------------------------+
| get_customer_balance(3, NOW()) |
+-----------------------------------------+
|                                  0.00 |
+-----------------------------------------+
1세트 1행(0.13초)
```

기한이 지난 DVD 찾기

많은 DVD 스토어에서 연체된 대여 목록을 매일 작성하여 고객에게 연락하여 연체된 DVD를 반납하도록 요청할 수 있습니다.

이러한 목록을 만들려면 대여 테이블에서 반납 날짜가 NULL이고 대여 날짜가 영화 테이블에 지정된 대여 기간보다 더 지난 영화를 검색합니다. 그렇다면 연체된 영화이므로 고객 이름 및 전화번호와 함께 영화 이름을 생성해야 합니다.

```
mysql> SELECT CONCAT(customer.last_name, ', ', customer.first_name) AS customer, address.phone, film.title
               FROM rental INNER JOIN customer ON rental.customer_id= customer.customer_id INNER JOIN address ON
               customer.address_id = address.address_id
               INNER JOIN 인벤토리에 렌탈.인벤토리_id ON= 인벤토리.인벤토리_id 인너 조인 필름에 인벤토리.필름_id = 필름.필름_id
               WHERE rental.return_date IS NULL
               AND rental_date+ INTERVAL film.rental_duration DAY< CURRENT_DATE() ORDER BY title
               제한 5;
+----------------+--------------+-----------------------------------+
| 고객                  | 전화               | title                   |
+----------------+--------------+-----------------------------------+
| 올베라, 드웨인 | 62127829280 | 아카데미 공룡 |
| 휴이, 브랜든            | 99883471275 | 에이스 골드핑거                 |
| 오웬스, 카멘            | 272234298332 | 불륜 편견 |
| 한논, 세스              | 864392582257 | 아프리카 달걀                 |
|                        | 371490777743 | ali forever
+-콜,--트레어사-----+--------------+-----------------------------------+
한 세트에 5줄(0.10초)
```

7 알려진 문제

Sakila 샘플 데이터베이스의 설계에서는 특정 스토어의 직원이 다른 스토어가 아닌 해당 스토어의 고객에게만 재고 품목을 대여한다고 가정합니다. 이 가정은 대여, 재고, 직원, 매장 테이블이 루프를 형성하는 관계를 갖는다는 점에서 분명하게 드러납니다. 고객은 하나의 스토어만 보유할 수 있지만 직원은 이와 비슷한 제약을 받지 않습니다. 직원이 다른 스토어에서 물품을 대여하는 경우 대여 테이블의 데이터가 일관되지 않을 수 있습니다.

이 문제에 대한 해결책은 독자의 몫입니다. 다음은 몇 가지 가능한 접근 방식입니다:

- 렌탈 테이블에 `store_id` 열을 추가하고 테이블의 외래 키도 해당 열을 참조하도록 하여 `customer_id` 및 `inventory_id`뿐만 아니라 재고 테이블의 `staff_id`도 동일한 스토어를 갖도록 합니다.
- 렌탈 테이블에 `INSERT` 및 `UPDATE` 트리거를 추가합니다.

8 감사

사킬라 샘플 데이터베이스의 초기 개발에 기여한 개인과 단체는 다음과 같습니다. 이 기록 목록은 더 이상 업데이트되지 않습니다.

- 롤랜드 부만: 개발 프로세스 전반에 걸쳐 귀중한 피드백을 제공하고, 샘플 보기와 저장 프로시저를 제공했습니다.
- 로널드 브래드포드: 사킬라 샘플 데이터베이스와 함께 사용할 수 있는 최초의 샘플 애플리케이션을 개발했습니다.
- Dave Jaffe: Dell 백서에 사용된 스키마를 제공하고 그 일부를 Sakila 샘플 데이터베이스에서 사용할 수 있는 권한을 확보했습니다.
- 주세페 막시아: 개발 프로세스 전반에 걸쳐 귀중한 피드백을 제공하고, 일부 샘플 데이터를 채우고, 일부 샘플 보기 및 트리거를 제공했습니다.

  v1.0의 경우 SQL 파일 내에 MySQL 버전별 주석을 추가하여 sakila와 결합했습니다.
- 제이 파이프: 몇 가지 샘플 저장 프로시저를 제공했습니다.
- Zak Greant: 라이선스에 대한 조언과 피드백을 제공했습니다.

앞서 언급한 개인 외에도 MySQL과 MySQL 커뮤니티의 다양한 사람들이 개발 과정에서 피드백을 제공했습니다.

9 사킬라 샘플 데이터베이스 라이선스

`sakila-schema.sql` 및 `sakila-data.sql` 파일의 콘텐츠는 New BSD 라이선스에 따라 라이센스가 부여됩니다.

새로운 BSD 라이선스에 대한 정보는 http://www.opensource.org/licenses/bsd-license.php 및 http://en.wikipedia.org/wiki/BSD_License 에서 확인할 수 있습니다.

이 설명서를 포함하여 Sakila 배포판에 포함된 추가 자료는 오픈 라이선스에 따라 라이선스가 부여되지 않습니다. 이 설명서의 사용은 법적 고지 사항에 설명된 약관의 적용을 받습니다.

자세한 내용은 http://www.mysql.com/about/contact/ 으로 문의하시기 바랍니다.

10 작성자를 위한 참고 사항

기사 및 서적에 Sakila 샘플 데이터베이스를 사용할 때는 예제에 사용된 Sakila 샘플 데이터베이스의 버전을 명시적으로 나열하는 것이 좋습니다. 이렇게 하면 독자가 동일한 버전을 다운로드하여 사용할 수 있으며 데이터 또는 스키마 업그레이드로 인해 발생할 수 있는 결과의 차이가 발생하지 않습니다.

11 사킬라 변경 내역

이 섹션에서는 사킬라 샘플 데이터베이스의 각 버전에서 변경된 사항에 대해 설명합니다.

- 버전 1.5
- 버전 1.4
- 버전 1.3

버전 1.5

- MySQL 버그 #112552 수정: 주소 필드에서 악센트 문자가 누락되었습니다.

버전 1.4

- MySQL 버그 #112131 수정: 다른 film_id 정의와 일치하도록 film_text.film_id 필드에 부호가 없는 필드를 만들었습니다.
- 배우가 없는 영화는 film_list 및 nicer_but_slower_film_list 보기에서 반환되지 않았습니다.

버전 1.3

- MySQL 버그 #106951을 수정했습니다: 도시 및 국가 필드에서 악센트 문자가 누락되었습니다. 해당 값은 세계 데이터베이스를 사용하여 업데이트되었습니다. 또한 악센트 문자 자체도 누락되었습니다.
- MySQL 버그 #107158을 수정했습니다: 결제 테이블에서 렌탈 아이디 값이 null인 5개의 행을 제거했습니다.

버전 1.2

- 데이터베이스 객체는 이제 `utf8`이 아닌 `utf8mb4`를 사용합니다. 이 변경으로 인해 MySQL 5.6에서 `VARCHAR(255)`로 선언된 `film.title` 열의 최대 키 길이가 767바이트인 오류가 발생했습니다(지정된 키가 너무 길다). 실제 최대 제목 길이는 27자이므로 최대 키 길이를 초과하지 않도록 열을 `VARCHAR(128)`로 다시 선언했습니다.
- `sakila-schema.sql` 및 `sakila-data.sql`에는 `SET NAMES utf8mb4` 문이 포함되어 있습니다.
- `sakila-data`.DOS(CRLF) 줄 끝에서 Unix(LF) 줄 끝으로 변환되었습니다.
- `address.location` 열은 `SPATIAL` 인덱스가 있는 `GEOMETRY` 열입니다. MySQL 8.0.3부터 인덱스 공간 열에 `SRID` 특성이 없는 한 `SPATIAL` 인덱스는 무시됩니다. MySQL 8.0.3 이상에서는 위치 열에 `SRID 0` 특성이 포함되도록 변경되었습니다.
- `staff.password` 열이 `VARCHAR(40) BINARY`로 선언되었습니다. 이는 `_bin` 콜레이션을 지정하기 위한 문자 열 선언에서 속기로

BINARY를 사용하는 것으로, 현재 더 이상 사용되지 않습니다.

MySQL 8.0.17. 이 열은 바이너리의 다시 선언되었습니다. 즉, VARCHAR(40) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin입니다.

- rewards_report() 저장 프로시저에서 min_dollar_amount_purchased 매개 변수가 DECIMAL(10,2) UNSIGNED로 선언되었습니다. DECIMAL과 함께 UNSIGNED의 사용은 MySQL에서 더 이상 사용되지 않습니다.
  8.0.17. 매개 변수가 UNSIGNED 없이 다시 선언되었습니다.

- film_in_stock() 및 film_not_in_stock() 저장 프로시저는 MySQL 8.0.17부터 더 이상 사용되지 않는 FOUND_ROWS() 함수를 사용했습니다. 각 프로시저에서 FOUND_ROWS() 쿼리는 연결된 쿼리와 동일한 FROM 및 WHERE 절과 함께 COUNT(*)를 사용하는 쿼리로 대체되었습니다. 이는 FOUND_ROWS()를 사용하는 것보다 비용이 더 많이 들지만 동일한 결과를 생성합니다.

- film_text 테이블은 이전 버전에서 테이블 생성 실패를 방지하기 위해 MySQL 5.6.10 이전 버전에서는 InnoDB가 아닌 MyISAM을 사용합니다. (그러나 여전히 MySQL 5.6.10 이상으로 업그레이드하는 것이 좋습니다.)

- 앞의 변경 사항에 따라 MySQL Workbench용 sakila.mwb 파일이 업데이트되었습니다.

버전 1.1

- 모든 MyISAM 참조를 제거했습니다. film_text 테이블과 그 FULLTEXT 정의는 이제 InnoDB를 사용합니다. 이전 MySQL 서버 버전(5.6.10 이하)을 사용하는 경우 MySQL을 업그레이드하는 것이 좋습니다. 업그레이드할 수 없는 경우 sakila-schema.sql SQL 파일에서 film_text 테이블의 ENGINE 값을 MyISAM으로 변경하세요.

버전 1.0

- MySQL 버전별 주석을 사용하여 sakila-schema.sql 및 sakila-spatial-schema.sql을 단일 파일로 병합했습니다.

  주소.location과 같은 공간 데이터는 MySQL 서버 기준으로 sakila 데이터베이스에 삽입됩니다.
  5.7.5(InnoDB에 공간 인덱싱 지원이 추가된 경우). 또한 사용되기 전인 MySQL 서버 5.6.10부터 InnoDB 전체 텍스트 검색이 사용됩니다.

버전 0.9

- 지오메트리 데이터 유형의 공간 데이터를 포함하는 사킬라 예제 데이터베이스의 추가 복사본이 추가되었습니다. 이 데이터베이스는 별도로 다운로드할 수 있으며, MySQL 서버 5.7.5 이상이 필요합니다. 이 데이터베이스를 사용하려면 sakila-schema.sql 파일 대신 sakila-spatial-schema.sql 파일을 로드하세요.

- MySQL 5.7.5부터 기본적으로 활성화된 ONLY_FULL_GROUP_BY SQL 모드와 호환되도록 nicer_but_slower_film_list 및 film_list 보기 정의의 GROUP BY 절을 수정했습니다.

버전 0.8

- film_id 값에 대한 변경 사항을 포함하도록 upd_film 트리거 정의를 수정했습니다.

- 액터_정보 보기를 추가했습니다.

- inventory_held_by_customer 함수에 대한 오류 처리기를 변경했습니다. 이제 함수에 암호화된 1329 대신 NOT FOUND에 대한 종료 핸들러가 추가되었습니다.

- 스키마 및 데이터 파일에 새로운 BSD 라이선스용 템플릿을 추가했습니다.

- MySQL 5.1에서 로딩을 지원하기 위해 적절한 경우 저장 프로시저 및 함수에 SQL 데이터 읽기를 추가했습니다.

- rewards_report 절차의 날짜 범위 문제를 수정했습니다(Goplat에게 감사드립니다).

버전 0.7

- 판매_별_스토어 보기에서 모든 스토어에 동일한 관리자가 나열되던 버그가 수정되었습니다.
- 함수가 여러 행을 반환하는 inventory_held_by_customer 함수의 버그를 수정했습니다.
- 데이터 로딩을 방해하지 않도록 rental_date 트리거를 sakila-data.sql 파일로 옮겼습니다.

버전 0.6

- film_in_stock 저장 프로시저를 추가했습니다.
- film_not_in_stock 저장 프로시저를 추가했습니다.
- 인벤토리_헬프_바이_고객 저장 함수를 추가했습니다.
- 인벤토리 in_stock 저장 함수를 추가했습니다.
- 로딩에 최적화된 데이터 파일(다중 행 INSERT 문, 트랜잭션). (고마워요 Giuseppe)
- 결제 테이블 로딩 스크립트에서 결제 금액이 무한대로 증가하던 오류가 수정되었습니다.

버전 0.5

- 제이 파이프가 제출한 판매_별_스토어 및 판매_별_필름 카테고리 뷰를 추가했습니다.
- Jay Pipes가 제출한 rewards_report 저장 프로시저를 추가했습니다.
- get_customer_balance 저장 프로시저를 추가했습니다.
- 샘플 데이터베이스에 데이터를 로드하기 위해 sakila-data.sql 파일을 추가했습니다.

버전 0.4

- 직원 테이블에 비밀번호 열을 추가했습니다(VARCHAR(40) BINARY 기본값 NULL).

버전 0.3

- address.district를 VARCHAR(20)으로 변경했습니다.
- customer.first_name을 VARCHAR(45)로 변경했습니다.
- customer.last_name을 VARCHAR(45)로 변경했습니다.
- customer.email을 VARCHAR(50)으로 변경했습니다.
- payment.rental_id 열(INT NULL 열)을 추가했습니다.
- payment.rental_id에 대한 외래 키가 rental.rental_id에 추가되었습니다.
- rental.rental_id 추가, INT 자동 증가, 대리 기본 만들어짐, 기존 기본 고유 키로 변경됨.

버전 0.2

- 모든 테이블에는 기존 동작을 사용하는 마지막 업데이트 시간 스탬프 열이 있습니다(기본값은 현재_시간 스탬프 업데이트 시 현재_시간 스탬프).

- actor_id는 이제 스몰린트입니다.
- address_id는 이제 SMALLINT입니다.
- category_id는 이제 TINYINT입니다.
- city_id는 이제 SMALLINT입니다.
- country_id는 이제 SMALLINT입니다.
- customer_id는 이제 SMALLINT입니다.
- 고객 테이블의 first_name, last_name은 이제 VARCHAR 대신 CHAR입니다.
- 고객 테이블에 이제 이메일 CHAR(50)이 있습니다.
- 고객 테이블의 create_date는 이제 DATETIME입니다(마지막 업데이트 타임스탬프를 수용하기 위해).
- customer 테이블에는 create_date 열이 NOW()로 설정되도록 강제하는 새로운 ON INSERT 트리거가 있습니다.
- film_id는 이제 SMALLINT입니다.
- 이제 film.description의 기본값은 NULL입니다.
- film.release_year에 연도 유형이 추가되었습니다.
- 언어 테이블과 함께 film.language_id 및 film.original_language_id가 추가되었습니다. 자막이 있는 외국 영화의 경우 original_language_id는 NULL, language_id는 NULL이 아닐 수 있습니다.
- film.length는 이제 SMALLINT입니다.
- film.category_id 열이 제거되었습니다.
- 새 테이블: film_category; 영화당 여러 카테고리를 허용합니다.
- film_text.category_id 열이 제거되었습니다.
- 인벤토리_id는 이제 MEDIUMINT입니다.
- payment_id는 이제 SMALLINT입니다.
- payment.payment_date는 이제 DATETIME입니다.
- 결제 테이블에 트리거를 추가하여 INSERT 시 payment_date가 NOW()로 설정되도록 합니다.
- rental.이제 rental.이제 DATETIME입니다.
- 렌탈 테이블에 트리거를 추가하여 INSERT 시 렌탈 NOW()로 설정되도록 합니다.
- staff_id는 이제 TINYINT입니다.
- staff.email 추가 (VARCHAR(50)).
- staff.username 추가 (VARCHAR(16)).
- store_id는 이제 TINYINT입니다.

- 새로운 `film_category` 테이블을 처리하도록 `film_list` 보기가 업데이트되었습니다.

- 새로운 영화_카테고리 테이블을 처리하도록 `nicer_but_slower_film_list` 뷰가 업데이트되었습니다.