Music Label Database – project documentation

Author: Mariusz Galjan

Date: 14.12.2019

Table of Contents:

1 Introduction

This is a database containing information that could be used by a music label to store information about the “assets” and collaborators they possess such as bands, albums, studios and producers. The database was implemented using SQL Anywhere Developer Edition. It contains various procedures, functions and triggers.

2 Problem description

Music labels often work with hundreds of bands at once. Over the years of its existence, such a label can accumulate in its portfolio thousands of bands, albums, tours, etc. Therefore, there is a need to create a database to manage these resources efficiently.

3 Project assumptions

3.1 Business description.

The record label works with various bands. Each band has a contract with the label. In addition, each band has a certain number of musicians to pay the appropriate remuneration. The label makes money mainly from the sale of albums created by its bands. Each album consists of a number of tracks and a title. The bands work with the producer to create the album. To record an album, a band must have a rented studio.

In addition, the label can make money from organizing concerts in different cities. Each such concert must have a fixed date and the number of tickets available for sale. Concerts can be combined into tours.

In the event of a breakdown, the team may report a need for new equipment.

3.2 Description by method of nouns and verbs.

The team has a signed contract.

The band releases an album.

The producer is working on the album.

The album consists of tracks.

The album is recorded in the studio.

The team has a rented studio.

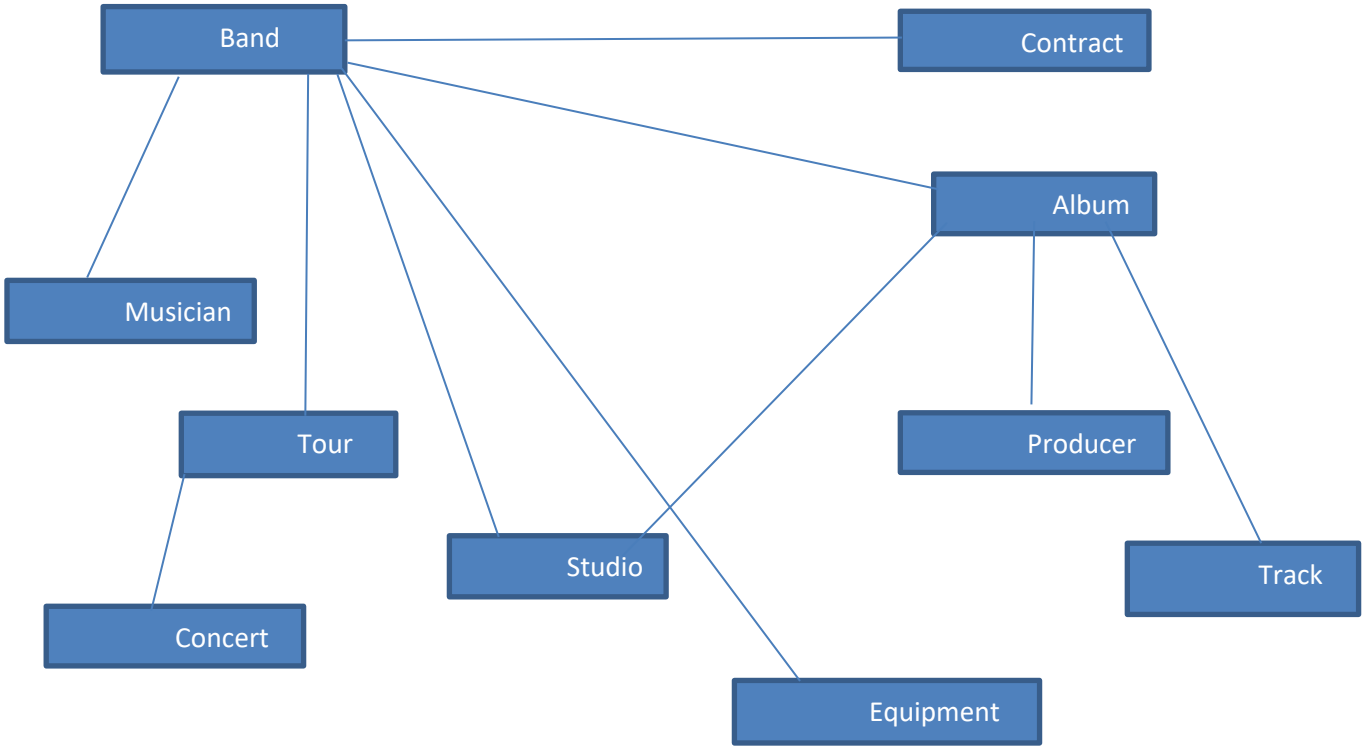
The band plays a tour.

The tour consists of concerts.

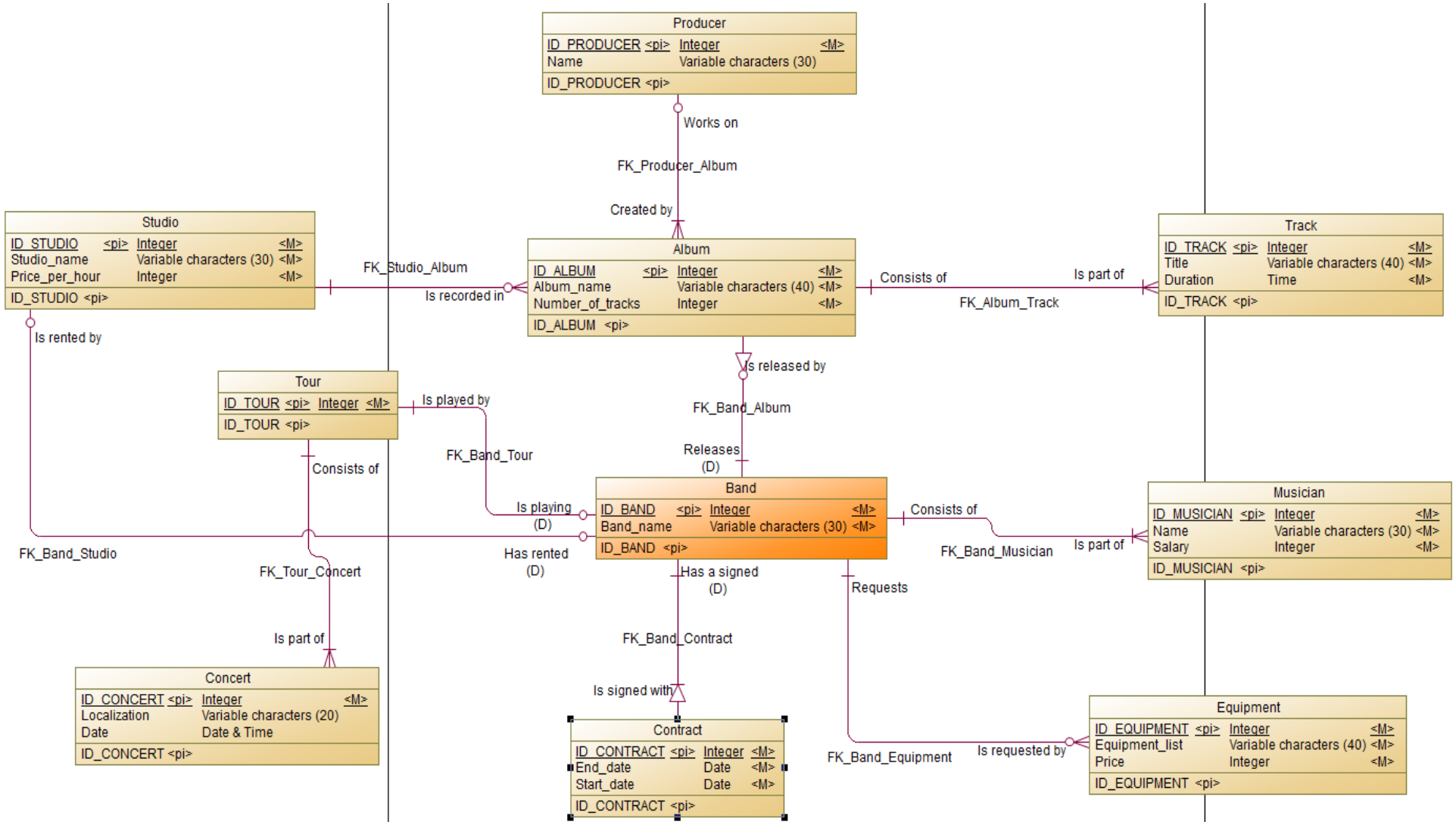
The musician belongs to the band.

The team is requesting equipment.

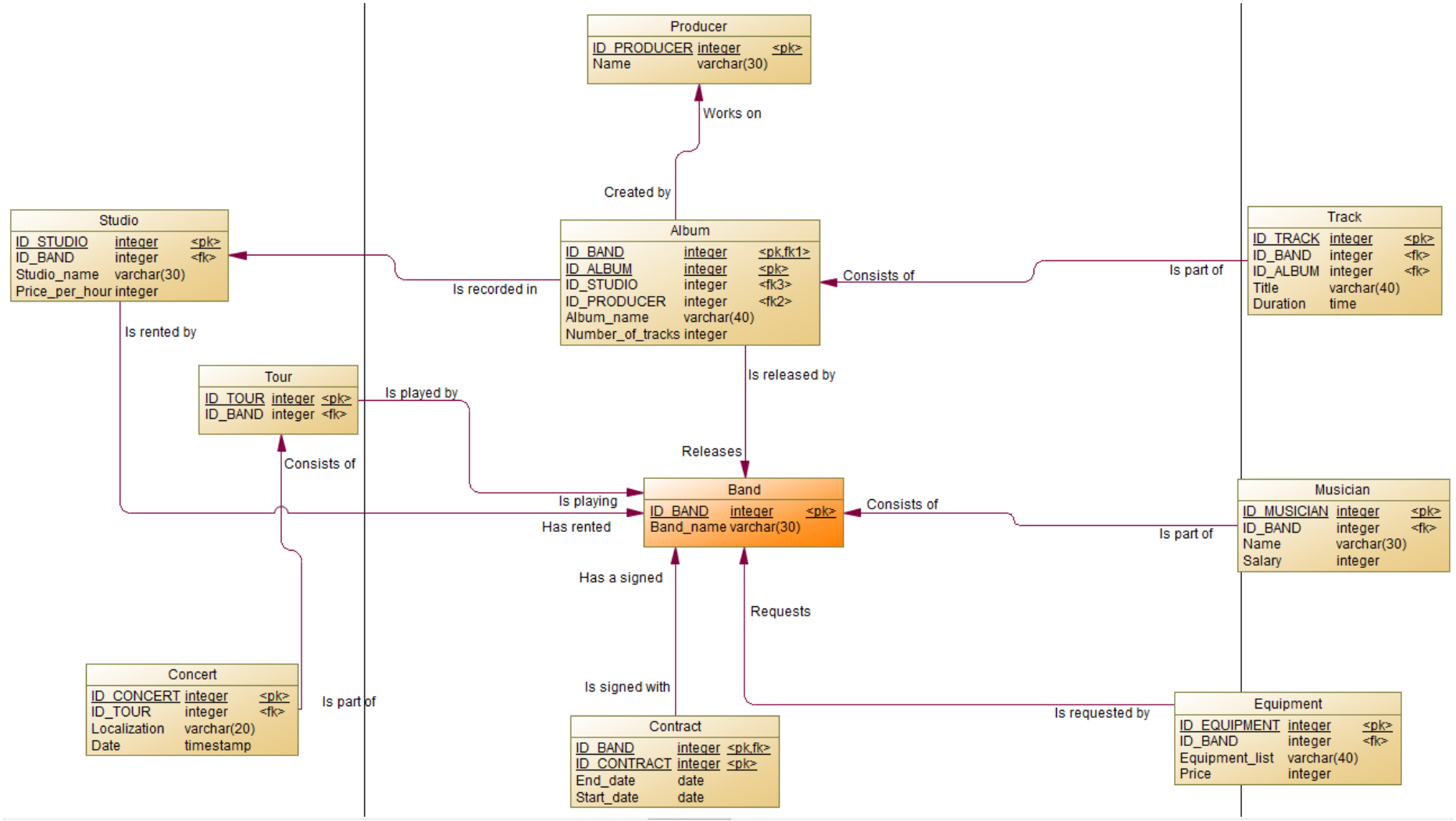
3.3 Relationships model.



4 Conceptual Database Model (CDM)



5 Physical Database model (PDM)



6 Create the database

5.1 Create a database structure.

Database structure creation scripts can be found in music_label_db_create.sql file.

6.2 Data entry.

Example data entry scripts can be found in the data_insert.sql

6.3 Listings of entered data (alphabetically)

Table Album:

	ID_BAND	ID_ALBUM	ID_STUDIO	ID_PRODUCER	Album_name	Number_of_tracks
1	1	1	1	2	Led Zeppelin I	9
2	3	2	5	1	Secret Place	4

Table Band:

	ID_BAND	Band_name
1	1	Led Zeppelin
2	2	The Beatles
3	3	MID
4	4	The White Stripes

Table Concert:

	ID_CONCERT	ID_TOUR	Localization	Date
1	1	1	Los Angeles	30.11.2019 00:00
2	2	1	Wembley	08.12.2019 00:00
3	3	1	Stadion Narodowy	15.12.2019 00:00
4	4	2	Detroit	10.03.2020 00:00
5	5	3	Liverpool	15.12.2019 00:00
6	6	3	Wembley	08.12.2019 00:00
7	7	4	OFFSIDE	22.11.2019 00:00
8	8	5	Sydney	10.01.2020 00:00
9	9	5	Tokyo	14.01.2020 00:00
10	10	6	Paryz	05.06.2020 00:00

Table Contract:

	ID_BAND	ID_CONTRACT	End_date	Start_date
1	1	1	01.01.2019	01.01.2012
2	2	2	12.03.2020	12.04.2017
3	3	3	21.06.2022	11.10.2015
4	4	4	30.12.2019	30.06.2013

Table Equipment:

	ID_EQUIPMENT	ID_BAND	Equipment_list	Price
1	1	2	Nowy wzmacniacz gitarowy	1 500
2	2	3	Mikrofon pojemnosciowy	800
3	3	3	Nowy wzmacniacz gitarowy	1 500

Table Musician:

	ID_MUSICIAN	ID_BAND	Name	Salary
1	1	1	John Bohnam	30 000
2	2	1	John Paul Jones	35 000
3	3	1	Jimmy Page	50 000
4	4	1	Robert Plant	30 000
5	5	2	Ringo Starr	60 000
6	6	2	Paul McCartney	100 000
7	7	2	John Lennon	100 000
8	8	2	George Harrison	60 000
9	9	3	Mariusz Galjan	120 000
10	10	3	Maciej Zalewski	120 000
11	11	3	Szymon Rapala	120 000
12	12	3	Krzysztof Grzelak	120 000
13	13	4	Jack White	80 000
14	14	4	Meg White	70 000

Table Producer:

	ID_PRODUCER	Name
1	1	Tim Berg
2	2	Steve McQueen
3	3	Warren Huart

Table Studio:

	ID_STUDIO	ID_BAND	Studio_name	Price_per_hour
1	1	1	Sunset Studios	100
2	2	2	Abbey Road Studios	150
3	3	1	Atlanta Studios	80
4	4	4	Universal Studios	90
5	5	3	Hear Studio	50

Table Tour:

	ID_TOUR	ID_BAND
1	1	1
2	2	1
3	3	2
4	4	3
5	5	4
6	6	4

Table Track:

	ID_TRACK	ID_BAND	ID_ALBUM	Title	Duration
1	1	1	1	Good Times Bad Times	00:02
2	2	1	1	Babe Im Gonna Leave You	00:06
3	3	1	1	You Shook Me	00:06
4	4	1	1	Dazed And Confused	00:06
5	5	1	1	Your Time Is Gonna Come	00:04
6	6	1	1	Black Mountain Side	00:02
7	7	1	1	Communication Breakdown	00:02
8	8	1	1	I Cant Quit You Baby	00:04
9	9	1	1	How Many More Times	00:08
10	10	3	2	Atlantica	00:04
11	11	3	2	Homecoming	00:04
12	12	3	2	Fleetwood	00:05
13	13	3	2	Secret Place	00:04

7 Queries (sorted by difficulty)

7.1 Show studios cheaper than 100 $.

```
/* Shows studios where the price per hour is smaller than 100$ */
SELECT Studio_name
    FROM Studio
    WHERE Price_per_hour < 100
```

Result:

	Studio_name
1	Atlanta Studios
2	Universal Studios
3	Hear Studio

7.2 Show Led Zeppelin tours

```
/* shows Led Zeppelin tours */
SELECT ID_TOUR
    FROM TOUR
    WHERE ID_BAND = 1
```

Result:

	ID_TOUR
1	1
2	2

7.3 Show tracks longer than 5 minutes

```
/* show tracks longer than 5 minutes */
SELECT Title
    FROM Track
    WHERE Duration >= '00:05'
```

Result:

	Title
1	Babe Im Gonna Leave You
2	You Shook Me
3	Dazed And Confused
4	How Many More Times
5	Fleetwood

7.4 Show musician names and the bands they play in

```
/* show musician names and the bands the play in. */
SELECT ID_MUSICIAN, Band.Band_name, Name
    FROM Musician
    JOIN Band ON  Band.ID_BAND = Musician.ID_BAND
```

Results:

	ID_MUSICIAN	Band_name	Name
1	1	Led Zeppelin	John Bohnam
2	2	Led Zeppelin	John Paul Jones
3	3	Led Zeppelin	Jimmy Page
4	4	Led Zeppelin	Robert Plant
5	5	The Beatles	Ringo Starr
6	6	The Beatles	Paul McCartney
7	7	The Beatles	John Lennon
8	8	The Beatles	George Harrison
9	9	MID	Mariusz Galjan
10	10	MID	Maciej Zalewski
11	11	MID	Szymon Rapala
12	12	MID	Krzysztof Grzelak
13	13	The White Stripes	Jack White
14	14	The White Stripes	Meg White

7.5 Show average track duration from each album with album name and its ID

```
/* show average track duration from each album with album name and its ID */
SELECT Album.ID_ALBUM, Album.Album_name, convert(time, DATEADD(ms, AVG(DATEDIF
F(ms, '00:00:00.000', Duration)), '00:00:00.000')) as Duration
    FROM Track
    JOIN Album ON Album.ID_ALBUM = Track.ID_ALBUM
    GROUP BY Album.ID_ALBUM, Album_name
```

Result:

	ID_ALBUM	Album_name	Duration
1	2	Secret Place	00:04
2	1	Led Zeppelin I	00:05

7.6 Show the numer of concerts each tour consists of

```
/* 6.6 show the number of concerts a tour consists of */
SELECT Tour.ID_TOUR, count(Concert.ID_CONCERT) as Number_of_concerts
    FROM Tour
    JOIN Concert ON Tour.ID_TOUR = Concert.ID_TOUR
    GROUP BY Tour.ID_TOUR
    ORDER BY Tour.ID_TOUR ASC
```

Results:

	ID_TOUR	Number_of_concerts
1	1	3
2	2	1
3	3	2
4	4	1
5	5	2
6	6	1

7.7 Show bands with more than 5 tracks composed

```
/* show names of bands with more than 5 tracks together with the numbers of th
ose tracks */
SELECT Band.ID_BAND, Band_name, count(Track.ID_BAND) AS number_of_tracks
    FROM Band
    JOIN Track ON Track.ID_BAND = Band.ID_BAND
    WHERE Band.ID_BAND IN (
        SELECT ID_BAND
            FROM Track
            GROUP BY ID_BAND
            HAVING  count(*) > 5
    )
    GROUP BY Band.ID_BAND, Band_name;
```

Results:

	ID_BAND	Band_name	number_of_tracks
1	1	Led Zeppelin	9

7.8 Show producer IDs, their names and tracks they produced

```
/* 6.8 show producer IDs, their names and tracks they worked on  */
SELECT Producer.ID_PRODUCER, Name, Track.Title
    FROM Producer, (
        SELECT ID_PRODUCER, Track.Title
            FROM Album
            JOIN Track ON Album.ID_ALBUM = Track.ID_ALBUM
    ) AS Track
    WHERE Producer.ID_PRODUCER = Track.ID_PRODUCER
    ORDER BY Producer.ID_PRODUCER ASC
```

Results:

	ID_PRODUCER	Name	Title
1	1	Tim Berg	Atlantica
2	1	Tim Berg	Fleetwood
3	1	Tim Berg	Homecoming
4	1	Tim Berg	Secret Place
5	2	Steve McQueen	You Shook Me
6	2	Steve McQueen	Dazed And Confused
7	2	Steve McQueen	Black Mountain Side
8	2	Steve McQueen	How Many More Times
9	2	Steve McQueen	Good Times Bad Times
10	2	Steve McQueen	I Cant Quit You Baby
11	2	Steve McQueen	Babe Im Gonna Leave You
12	2	Steve McQueen	Communication Breakdown
13	2	Steve McQueen	Your Time Is Gonna Come

7.9 Show bands and the number of concerts they are playing, sorted by the number of concerts descending

```
/* 6.9 show bands and the number of their concerts, sorted by the concerts_num
ber descending */
SELECT Band.Band_name, sum(Concerts.Number_of_concerts) AS Concerts
    FROM Band, (
        SELECT Tour.ID_BAND, Tour.ID_TOUR, count(Concert.ID_CONCERT) as Number
_of_concerts
            FROM Tour
            JOIN Concert ON Tour.ID_TOUR = Concert.ID_TOUR
            GROUP BY Tour.ID_TOUR, Tour.ID_BAND
            ORDER BY Tour.ID_TOUR ASC
    ) AS Concerts
    WHERE Band.ID_BAND = Concerts.ID_BAND
    GROUP BY Band.Band_name
    ORDER BY Concerts DESC
```

Results:

	Band_name	Concerts
1	Led Zeppelin	4
2	The White Stripes	3
3	The Beatles	2
4	MID	1

8 Views

8.1 View of studios with a price per hour smaller than 100 $

View:

	Studio_name
1	Atlanta Studios
2	Universal Studios
3	Hear Studio

8.2 View of bands and the number of concerts they are playing, sorted by the number of concerts descending

```
/* show bands and the number of their concerts, sorted by the concerts_number
descending */
CREATE VIEW Band_concerts AS
    SELECT Band.Band_name, sum(Concerts.Number_of_concerts) AS Concerts
    FROM Band, (
        SELECT Tour.ID_BAND, Tour.ID_TOUR, count(Concert.ID_CONCERT) as Number
_of_concerts
            FROM Tour
            JOIN Concert ON Tour.ID_TOUR = Concert.ID_TOUR
            GROUP BY Tour.ID_TOUR, Tour.ID_BAND
            ORDER BY Tour.ID_TOUR ASC
    ) AS Concerts
    WHERE Band.ID_BAND = Concerts.ID_BAND
    GROUP BY Band.Band_name
    ORDER BY Concerts DESC
```

View:

	Band_name	Concerts
1	Led Zeppelin	4
2	The White Stripes	3
3	The Beatles	2
4	MID	1

8.3 View of number of concerts per tour

```sql
/* 7.3  View of number of concerts per tour */
CREATE VIEW Concert_of_tour AS
    SELECT Tour.ID_TOUR, count(Concert.ID_CONCERT) as Number_of_concerts
    FROM Tour
    JOIN Concert ON Tour.ID_TOUR = Concert.ID_TOUR
    GROUP BY Tour.ID_TOUR
    ORDER BY Tour.ID_TOUR ASC
```

View:

	ID_TOUR	Number_of_concerts
1	1	3
2	2	1
3	3	2
4	4	1
5	5	2
6	6	1

9 Procedures

9.1 Procedure adding to the database a band that is not present in the database

```
/* Procedure adding to the database a band that is not present in the database
 */
ALTER PROCEDURE "DBA"."addBand"(IN bandId INTEGER , IN bandName VARCHAR(30) )
BEGIN
    IF NOT EXISTS (SELECT Band_name FROM Band WHERE Band_name = bandName) THEN
    BEGIN
        INSERT INTO Band VALUES
            (bandId, bandName);
    END
    ELSE BEGIN
        RAISERROR 99999 'Band already exists'
    END
    ENDIF;
END
```

Before:

	ID_BAND	Band_name
1	1	Led Zeppelin
2	2	The Beatles
3	3	MID
4	4	The White Stripes

After command ‘CALL "dba".addBand("bandId" = 6, "bandName" = 'The Jets')’:

	ID_BAND	Band_name
1	1	Led Zeppelin
2	2	The Beatles
3	3	MID
4	4	The White Stripes
5	6	The Jets

9.2 Procedure deleting a record of ID_CONCERT field equal to concertId from Concert table

```
ALTER PROCEDURE "dba"."deleteConcert"( IN concertId INTEGER )
BEGIN
    IF NOT EXISTS (SELECT ID_CONCERT FROM Concert WHERE ID_CONCERT = concertId
) THEN
        BEGIN
            RAISERROR 99999 'No record found'
        END
    ELSE
        BEGIN
            DELETE FROM Concert WHERE DBA.Concert.ID_CONCERT = concertId
        END
    ENDIF
END
```

Before:

	ID_CONCERT	ID_TOUR	Localization	Date
1	1	1	Los Angeles	30.11.2019 00:00
2	2	1	Wembley	08.12.2019 00:00
3	3	1	Stadion Narodowy	15.12.2019 00:00
4	4	2	Detroit	10.03.2020 00:00
5	5	3	Liverpool	15.12.2019 00:00
6	6	3	Wembley	08.12.2019 00:00
7	7	4	OFFSIDE	22.11.2019 00:00
8	8	5	Sydney	10.01.2020 00:00
9	9	5	Tokyo	14.01.2020 00:00
10	10	6	Paryz	05.06.2020 00:00

After ‘CALL "dba"."deleteConcert"("concertId" = 4)’:

	ID_CONCERT	ID_TOUR	Localization	Date
1	1	1	Los Angeles	30.11.2019 00:00
2	2	1	Wembley	08.12.2019 00:00
3	3	1	Stadion Narodowy	15.12.2019 00:00
4	5	3	Liverpool	15.12.2019 00:00
5	6	3	Wembley	08.12.2019 00:00
6	7	4	OFFSIDE	22.11.2019 00:00
7	8	5	Sydney	10.01.2020 00:00
8	9	5	Tokyo	14.01.2020 00:00
9	10	6	Paryz	05.06.2020 00:00

9.3 Procedure deleting a record of ID_TOUR field equal to tourId from Tour table and concerts from that tour from the Concert table

```
ALTER PROCEDURE "dba"."deleteTour"( IN tourId INTEGER )
BEGIN
    IF NOT EXISTS (SELECT ID_TOUR FROM Tour WHERE ID_TOUR = tourId) THEN
        BEGIN
            RAISERROR 99999 'No record found'
        END
    ELSE
        BEGIN
            DELETE FROM Concert WHERE DBA.Concert.ID_TOUR = tourId;
            DELETE FROM Tour WHERE DBA.Tour.ID_TOUR = tourId;
        END
    ENDIF
END
```

Before:

	ID_TOUR	ID_BAND
1	1	1
2	2	1
3	3	2
4	4	3
5	5	4
6	6	4

After:

	ID_TOUR	ID_BAND
1	1	1
2	2	1
3	4	3
4	5	4
5	6	4

10 Functions

10.1 Get salary of musician with ID_MUSICIAN = musicianId

```
/* 9.1 Get salary of musician with ID_MUSICIAN = musicianId */
ALTER FUNCTION "DBA"."getSalary"( IN musicianId INTEGER )
RETURNS INTEGER
DETERMINISTIC
BEGIN
    DECLARE salary INTEGER;
    DECLARE musician INTEGER;
    SET musician = musicianId;
    SET salary = (SELECT Salary FROM Musician WHERE musician = Musician.ID_MUS
ICIAN);
    RETURN salary;
END
```

Results of calling 'SELECT "dba"."getSalary"(3)':

	dba.getSalary(3)
1	50 000

10.2 Get id of the most expensive studio thats within the budget

```
/* 9.2 Get id of the most expensive studio thats within the budget */
ALTER FUNCTION "dba"."studioInBudget"( IN budget INTEGER )
RETURNS INTEGER
DETERMINISTIC
BEGIN
    DECLARE "studioId" INTEGER;
    SET studioId = (SELECT ID_STUDIO
                        FROM (SELECT TOP 1 ID_STUDIO, Price_per_hour AS Price
                                FROM Studio
                                WHERE Price_per_hour < 100
                                ORDER BY Price DESC) AS StudioPrice );
    RETURN "studioId";
END
```

Result of 'SELECT "dba"."studioInBudget"(100)':

	dba.studioInBudget(100)
1	4

11 Cursors

11.1 Procedure printing band IDs with their names

```
/* 10.1 Procedure printing band IDs with their names */
ALTER PROCEDURE "dba"."showBands"()
BEGIN
    DECLARE i INTEGER;
    DECLARE bandId INTEGER;
    DECLARE bandName VARCHAR(30);
    DECLARE bandCursor DYNAMIC SCROLL CURSOR FOR
        SELECT ID_BAND, Band_name FROM Band;

    SET i = 0;
    OPEN bandCursor;
    WHILE i < (SELECT count(ID_BAND) FROM Band)
    LOOP
        FETCH NEXT bandCursor INTO bandId, bandName;
        MESSAGE 'ID: ', + bandId, + ' Band name: ', + bandName TO CLIENT;

        SET i = i + 1;
    END LOOP;
    CLOSE bandCursor;
    DEALLOCATE bandCursor;
END
```

Result:

```
CALL "dba"."showBands"()
ID: 1 Band name: Led Zeppelin
ID: 2 Band name: The Beatles
ID: 3 Band name: MID
ID: 4 Band name: The White Stripes
Procedure completed
```

11.2 Function returning number of planned concerts (using cursors)

```
/* 10.2 Function returning number of planned concerts */
ALTER FUNCTION "DBA"."numberOfPlannedConcerts"( )
RETURNS INTEGER
DETERMINISTIC
BEGIN
    DECLARE numberOfConcerts INTEGER;
    DECLARE i INTEGER;
    DECLARE tourId INTEGER;
    DECLARE concertsInTour INTEGER;
    DECLARE cursorVar DYNAMIC SCROLL CURSOR FOR
        /* selects number of concerts for each tour*/
        SELECT Tour.ID_TOUR, count(Concert.ID_CONCERT) as number_of_concerts
            FROM Tour
            JOIN Concert ON Tour.ID_TOUR = Concert.ID_TOUR
            GROUP BY Tour.ID_TOUR
            ORDER BY Tour.ID_TOUR ASC;
    SET numberOfConcerts = 0;

    SET i = 0;
    OPEN cursorVar;
    WHILE i < (SELECT count(Concerts.ID_TOUR) FROM (
                    /* counts number of tours in the Concerts table */
                    SELECT Tour.ID_TOUR, count(Concert.ID_CONCERT) as number_o
f_concerts
                        FROM Tour
                        JOIN Concert ON Tour.ID_TOUR = Concert.ID_TOUR
                        GROUP BY Tour.ID_TOUR ) AS Concerts )
    LOOP
        FETCH NEXT cursorVar INTO tourId, concertsInTour;
        SET numberOfConcerts = numberOfConcerts + concertsInTour;
        SET i = i + 1;
    END LOOP;
    CLOSE cursorVar;
    DEALLOCATE cursorVar;

    RETURN "numberOfConcerts";
END
```

Results:

	dba.numberOfPlannedConcerts()
1	10

11.3 Procedure printing producers with the tracks they worked on

```
/* Procedure printing producers with the tracks they worked on */
ALTER PROCEDURE "dba"."showProducersTracks"( )
BEGIN
    DECLARE i INTEGER;
    DECLARE producerId INTEGER;
    DECLARE actProducerId INTEGER;
    DECLARE producerName VARCHAR(30);
    DECLARE trackId INTEGER;
    DECLARE trackTitle VARCHAR(30);
    DECLARE cursorVar DYNAMIC SCROLL CURSOR FOR
        /* selects producer's ID, Name and the tracks he/she's been working on
 */
        SELECT Producer.ID_PRODUCER, Name, Track.Title
            FROM Producer, (
                SELECT ID_PRODUCER, Track.Title
                FROM Album
                JOIN Track ON Album.ID_ALBUM = Track.ID_ALBUM
            ) AS Track
            WHERE Producer.ID_PRODUCER = Track.ID_PRODUCER
            ORDER BY Producer.ID_PRODUCER ASC;

    SET actProducerId = 0;

    SET i = 0;
    OPEN cursorVar;
    WHILE i < (SELECT count(ProducentTrack.Title) FROM (
        SELECT Producer.ID_PRODUCER, Name, Track.Title
            FROM Producer, (
                SELECT ID_PRODUCER, Track.Title
                FROM Album
                JOIN Track ON Album.ID_ALBUM = Track.ID_ALBUM
            ) AS Track
            WHERE Producer.ID_PRODUCER = Track.ID_PRODUCER
            ORDER BY Producer.ID_PRODUCER ASC) AS ProducentTrack)
    LOOP
        FETCH NEXT cursorVar INTO producerId, producerName, trackTitle;

        /*If we fetched the next producer, we print his name before his tracks
 */
        IF actProducerId != producerId THEN
            BEGIN
                MESSAGE 'Producent: ', + producerName TO CLIENT;
                SET actProducerId = producerId;
            END
        ENDIF;

        MESSAGE trackTitle TO CLIENT;
```

```
        SET i = i + 1;
    END LOOP;
    CLOSE cursorVar;
    DEALLOCATE cursorVar;
END
```

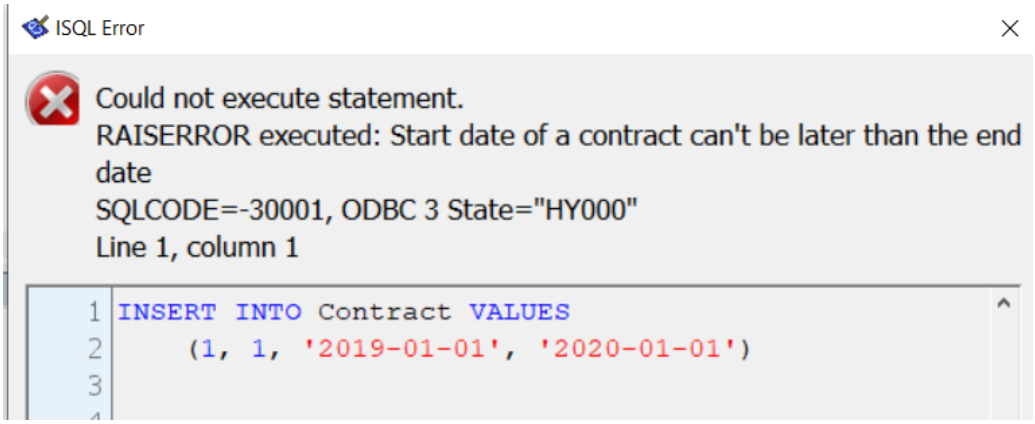
Results of ‘CALL "dba"."showProducersTracks"()’:

```
Producer: Tim Berg
Atlantica
Fleetwood
Homecoming
Secret Place
Producer: Steve McQueen
You Shook Me
Dazed And Confused
Black Mountain Side
How Many More Times
Good Times Bad Times
I Cant Quit You Baby
Babe Im Gonna Leave You
Communication Breakdown
Your Time Is Gonna Come
Procedure completed
```

12 Triggers

12.1 Insert trigger checking, if the start date of a contract is later than the end date

Result:



12.2 Insert trigger updating Album.Number_of_tracks field on adding a new track

```
/* 11.2 Insert trigger updating Album.Number_of_tracks field on adding a new t
rack */
CREATE TRIGGER "addTrack" AFTER INSERT
ORDER 1 ON "dba"."Track"
REFERENCING NEW AS newTrack
FOR EACH ROW
BEGIN
    DECLARE numberOfTracks INTEGER;

    SET numberOfTracks = (SELECT Number_of_tracks FROM Album WHERE ID_ALBUM =
newTrack.ID_ALBUM);
    MESSAGE 'Current number of tracks: ', + numberOfTracks TO CLIENT;

    SET numberOfTracks = numberOfTracks + 1;
    UPDATE Album
        SET Number_of_tracks = numberOfTracks
        WHERE ID_ALBUM = newTrack.ID_ALBUM;

    MESSAGE 'Updated number of tracks: ', + numberOfTracks TO CLIENT;
END
```

12.3 Delete trigger deleting concerts from tour when deleting a tour

```
/* 11.3 Delete trigger deleting concerts from tour when deleting a tour */
CREATE TRIGGER "deleteConcertsFromTour" BEFORE DELETE
ORDER 1 ON "dba"."Tour"
REFERENCING OLD AS tour
FOR EACH ROW
BEGIN
    DELETE FROM Concert WHERE Concert.ID_TOUR = tour.ID_TOUR;
    MESSAGE 'Deleted concerts from tour with ID ', + tour.ID_TOUR TO CLIENT;
END
```