Tittle:

SQL Programming Project ERD

Career:

Computer Systems Engineering

Students:

Daniel García de Luna: UP200423

Juan José Rojas Cornejo: UP200760

Juan Pablo Rodríguez Serna: UP200512

Luis Ernesto Pérez Lechuga: UP200966

José Ángel Silva Salgado: UP200511

Juan Pablo Alcalá López: UP200231

Topic:

Database Administration

Institution:

Universidad Politécnica de Aguascalientes

Group:

ISC06B

Place and date of delivery:

Universidad Politécnica de Aguascalientes, October 26, 2022.

INDEX

Introduction:
This document presents the development of an online media rental database by implementing Oracle and MySQL.

The different commands seen in class, the different statements and logic are taught by the students. This document also presents the explanation of how the database was created.

The database reflects the requirements specified by means of the evaluation rubric, such as restrictions, table creation, data insertion, sequences, as well as improvements that were considered to be more efficient.

The database is idealized to be used in movie rental stores or different services, an example of this would be Blockbuster (currently closed) in order to have a better organization of their information.

With this document, the files for user creation, table creation, data insertion and queries in Oracle and MySQL are also attached.

Development:

MySQL

To start the creation of the database in MySQL, we start by creating the user and together with this the permissions are provided.

```sql
CREATE DATABASE bussines;
CREATE USER 'admin'@'localhost' IDENTIFIED BY 'admin';
GRANT ALL PRIVILEGES ON *.* TO 'admin'@'localhost';
```

1 Code for Creation a main user in MySQL

After the creation of the user, we create the 'customers' table and this is assigned the types of variables according to the customer's requirements.

```sql
create table customers (
    `CUSTOMER_ID` INT(10) AUTO_INCREMENT NOT NULL PRIMARY KEY,
    `LAST_NAME` VARCHAR(25) NOT NULL,
    `FIRST_NAME` VARCHAR(25) NOT NULL,
    `HOME_PHONE` VARCHAR(12) NOT NULL,
    `ADDRESS` VARCHAR(100) NOT NULL,
    `CITY` VARCHAR(30) NOT NULL,
    `STATE` VARCHAR(2) NOT NULL,
    `EMAIL` VARCHAR(25),
    `CELL_PHONE` VARCHAR(12)
);
```

1.1 Creation of the ´customers´ table

The table of 'movies' is created along with their respective relations (constrains).

```sql
CREATE TABLE `MOVIES`
( `TITLE_ID` INT(10) PRIMARY KEY AUTO_INCREMENT NOT NULL,
`TITLE` VARCHAR(60) NOT NULL, `DESCRIPTION` VARCHAR(400) NOT NULL,
`RATING` VARCHAR(4), `CATEGORY` VARCHAR(20), `RELEASE_DATE` DATE NOT NULL,

CONSTRAINT `RATING_CHECK`CHECK(`RATING` IN ('G','PG','PG13','R')),
CONSTRAINT `CATEGORY_CHECK` CHECK (`CATEGORY` IN ('DRAMA','COMEDY','ACTION'
,'CHILD', 'SCIFY', 'DOCUMENTARY')) );
```

1.2 Creation of the ´movies´ table

The table of 'movies' is created along with their respective relations (constrains).

```sql
CREATE TABLE `MEDIA` (
    `MEDIA_ID` INT(10) PRIMARY KEY AUTO_INCREMENT NOT NULL,
    `FORMAT` VARCHAR(3) NOT NULL,
    `TITLE_ID` INT(10) NOT NULL,
     CONSTRAINT FK_TITLE FOREIGN KEY (`TITLE_ID`) REFERENCES MOVIES(`TITLE_
ID`)
);
```

1.3 Creation of the ´mean´ table

```sql
CREATE TABLE `RENTAL_HISTORY`( `MEDIA_ID` INT(10) NOT NULL, `RENTAL_DATE` D
ATE NOT NULL DEFAULT CURRENT_TIMESTAMP, `CUSTOMER_ID` INT(10) NOT NULL, `RE
TURN_DATE` DATE,
    CONSTRAINT `FK_CUSTOMER` FOREIGN KEY (`CUSTOMER_ID`) REFERENCES custome
rs(`CUSTOMER_ID`),
    CONSTRAINT PK_COMP PRIMARY KEY (`MEDIA_ID`, `RENTAL_DATE`) );
```

1.4 Creation of the ´rental_history´ table

```sql
CREATE TABLE `ACTORS` (
    `ACTOR_ID` INT(10) PRIMARY KEY AUTO_INCREMENT NOT NULL,
    `STAGE_NAME` VARCHAR(40) NOT NULL,
    `FIRST_NAME` VARCHAR(25) NOT NULL,
    `LAST_NAME` VARCHAR(25) NOT NULL,
    `BIRTH_NAME` DATE NOT NULL
);
```

1.5 Creation of the ´actors´ table

```sql
CREATE TABLE `SATR_BILLINGS` (
    `ACTOR_ID` INT(10) NOT NULL,
    `TITLE_ID` INT(10) NOT NULL,
    `COMMENTS` VARCHAR(40),
     CONSTRAINT PK_COMP PRIMARY KEY (`ACTOR_ID`, `TITLE_ID`)
);
```

1.6 Creation of the table ´sart_billings´

We created a view to know the titles that are not available.

```sql
CREATE VIEW  `TITLE_UNAVAIL` AS
    SELECT RE.MEDIA_ID, MO.TITLE FROM MOVIES MO, RENTAL_HISTORY RE, MEDIA M
E
    WHERE MO.TITLE_ID = ME.TITLE_ID AND ME.MEDIA_ID = RE.MEDIA_ID AND RE.RE
TURN_DATE IS NULL;
```

1.7 Creation of the ´title_unavail´ view

We create in the tables the 'alter table' to establish the auto increments on each of the needs of the tables.

```sql
ALTER TABLE customers AUTO_INCREMENT=101;

ALTER TABLE movies AUTO_INCREMENT=1;

ALTER TABLE media AUTO_INCREMENT=92;

ALTER TABLE `actors` CHANGE `ACTOR_ID` `ACTOR_ID` INT(10) NOT NULL AUTO_INC
REMENT;

ALTER TABLE actors AUTO_INCREMENT=1001;
```

1.8 Creation of the 'alter table'

We create the index and test the index to verify.

```sql
CREATE INDEX INDEX_LN ON `CUSTOMERS`(LAST_NAME);

SHOW INDEX FROM CUSTOMERS;
```

1.9 Creating and checking the ‘index’

We check the tables to conclude.

```sql
SELECT * FROM ACTORS;

SELECT * FROM CUSTOMERS;

SELECT * FROM MEDIA;

SELECT * FROM MOVIES;

SELECT * FROM RENTAL_HISTORY;

SELECT * FROM STAR_BILINGS;
```

Image 1.9: Checking the tables

Oracle

We started creating the user who is going to manage the database.

```sql
CREATE USER OracleFlix
IDENTIFIED BY OracleFlix
default tablespace users
temporary tablespace temp
quota unlimited on users;
```

2.1 User creation

We granted privileges to the user, to manage the database.

```sql
GRANT CONNECT, RESOURCE, CREATE VIEW TO OracleFlix;
grant create any index to OracleFlix;
GRANT EXECUTE any PROCEDURE TO OracleFlix;
```

2.2 Grant privileges

Then, we proceed to create the specified tables, with its respective constraints.

- Customers

```sql
CREATE TABLE customers
            (customer_id NUMBER(10)constraint customer_id_pk not null enable,
            last_name VARCHAR2(25) NOT NULL enable,
            first_name VARCHAR2(25) NOT NULL enable,
            home_phone VARCHAR2(12) NOT NULL enable,
            address VARCHAR2(100) NOT NULL enable,
            city VARCHAR2(30) NOT NULL enable,
            state VARCHAR2(2) NOT NULL enable,
            email VARCHAR2(25),
            cell_phone VARCHAR2(12),
            primary key (customer_id));
```

2.3 Customers table creation

- Movies

```sql
CREATE TABLE movies
            (title_id NUMBER(10) CONSTRAINT title_id_pk NOT NULL ENABLE,
            title VARCHAR2(60) NOT NULL ENABLE,
            description VARCHAR2(400) NOT NULL ENABLE,
            rating VARCHAR2(4) CONSTRAINT movies_rating CHECK (rating IN ('G', 'PG','R','PG13')),
            category VARCHAR2(20) CHECK (category IN ('DRAMA', 'COMEDY', 'ACTION', 'CHILD', 'SCIFI', 'DOCUMENTARY')),
            release_date date NOT NULL ENABLE,
            primary key (title_id));
```

2.4 Movies table creation

- Media

```sql
CREATE TABLE media
            (media_id NUMBER(10) CONSTRAINT media_id_pk PRIMARY KEY,
                format VARCHAR2(3) NOT NULL ENABLE,
                title_id NUMBER(10) NOT NULL ENABLE CONSTRAINT media_titleid_fk REFERENCES movies(title_id));
```

2.5 Media table creation

- Rental_history

```sql
CREATE TABLE rental_history
                (media_id NUMBER(10) CONSTRAINT media_id_fk NOT NULL ENABLE REFERENCES media(media_id),
            rental_date date DEFAULT sysdate not null enable,
                customer_id NUMBER(10) CONSTRAINT customer_id_fk NOT NULL ENABLE REFERENCES customers(customer_id),
                return_date date,
                CONSTRAINT rental_history_pk PRIMARY KEY(media_id, rental_date));
```

2.6 Rental_history table creation

- Actors

```
CREATE TABLE actors (actor_id NUMBER(10) CONSTRAINT actor_id_pk NOT NULL ENABLE,
                stage_name VARCHAR2(40) NOT NULL ENABLE,
                last_name VARCHAR2(25) NOT NULL ENABLE,
                first_name VARCHAR2(25) NOT NULL ENABLE,
                birth_date date NOT NULL ENABLE,
                PRIMARY KEY(actor_id));
```

2.7 Actors table creation

- Star_billings

```
CREATE TABLE star_billings
        (actor_id NUMBER(10) CONSTRAINT actor_id_fk NOT NULL ENABLE REFERENCES actors(actor_id),
        title_id NUMBER(10) CONSTRAINT title_id_fk NOT NULL ENABLE REFERENCES movies(title_id),
        comments VARCHAR2(40),
        CONSTRAINT star_billings_pk PRIMARY KEY (actor_id, title_id));
```

2.8 Star_billings table creation

The we create the sequences, to auto increment the primary keys.

- Customers

```
create sequence customer_id_seq
                INCREMENT by 1
                start with 101;
```

2.9 Customer sequence creation

- Movies

```
create sequence title_id_seq
                INCREMENT by 1
                start with 1;
```

2.10 Movies sequence creation

- Media

```
create sequence media_id_seq
                INCREMENT by 1
                start with 92;
```

2.11 Media sequence creation

- Actor

```
create sequence actor_id_seq
                increment by 1
                start with 1001;
```

2.12 Actor sequence creation

Finally, the data insertion, it’s important to make it in order, the first ones are customers, movies and actors, because they don’t have foreign keys. Then we insert the data to star_billings, and media, cause this ones have foreign keys of the previous tables, and then, the rental_history, because this one has a foreign key that depends from the table media.

- Customers

```
insert into customers
            values (customer_id_seq.nextval, 'Palombo', 'Lisa', '716-270-2669', '123 Main St', 'Buffalo', UPPER('NY'), 'palombo@ecc.edu', '716-555-1212');
        insert into customers
            values (customer_id_seq.nextval, 'Lulu', 'Liz', '716-278-4569', '14 San Ge', 'Aguascalientes', upper('ag'), 'Lulu@ecc.edu', '456-555-1982');
        insert into customers
            values (customer_id_seq.nextval, 'Polu', 'Lucho', '986-458-1119', '94 Lent Ls', 'Mexico', upper('ME'), 'Polu@ecc.eduu', '686-524-782');
        insert into customers
            values (customer_id_seq.nextval, 'Rojas', 'Juan', '449-804-4717', '201 Peñuelas AGS', 'Aguascalientes', UPPER('ag'), 'juan@ecc.edu', '659-784-6589');
        insert into customers
            values (customer_id_seq.nextval, 'Gallegos', 'Vivian', '449-887-9527', '265 Casa Solida AGS', 'Aguascalientes', upper('AG'), 'viv@ecc.edu', '548-798-6125');
        insert into customers
            values (customer_id_seq.nextval, 'Gonzales', 'Israel', '789-567-9854', '658 Mexico AGS', 'Aguascalientes', upper('AG'), 'isra@ecc.edu', '58-998-6178');
```

2.13 Customers data insertion

- Movies

```
insert into movies
            values(MOVIE_ID_SEQ.nextval, 'Remember the Titans', 'The true story of a newly appointed African-American
            coach and his high school team on their first season as a
            racially integrated unit.', 'PG', 'DRAMA', TO_DATE('29/09/2000', 'dd/mm/yyyy') );
        insert into movies
            values(title_ID_SEQ.nextval, 'Black Adam','Dwayne Johnson brings the anti-hero Black Adam to the DC Extended Universe this fall.',
            UPPER('PG13'), UPPER('action'), TO_DATE('20/10/2020', 'dd/mm/yyyy') );
        insert into movies
            values(title_ID_SEQ.nextval, '"Pennyworth"','Set against the colorful backdrop of 1960s London, "Pennyworth" explores the gritty origin story of Bruce Waynes legendary butler
            (and former bad ass British SAS soldier), Alfred Pennyworth.',

            UPPER('PG13'), UPPER('action'), TO_DATE('15/04/2024', 'dd/mm/yyyy') );
        insert into movies
            values(title_ID_SEQ.nextval, '"Stargirl"','Were always excited when a new hero is added to the CWs pantheon and we loved seeing Stargirl become yet another addition to
            the ever-growing Arrowverse crossover events when the show premiered in 2020.',
            UPPER('R'), UPPER('action'), TO_DATE('30/10/2028', 'dd/mm/yyyy') );
        insert into movies
            values(title_ID_SEQ.nextval, '"Stargirl"','Were always excited when a new hero is added to the CWs pantheon and we loved seeing Stargirl become yet another addition to
            the ever-growing Arrowverse crossover events when the show premiered in 2020.',
            UPPER('R'), UPPER('action'), TO_DATE('30/10/2028', 'dd/mm/yyyy') );
        insert into movies
            values(title_ID_SEQ.nextval, '"The Sandman"','Neil Gaiman’s groundbreaking comic series "The Sandman" is finally coming to our screens, after having been thought of as "unadaptable" for decades.
            Fans are looking forward to a modern retelling of the source material ',
            UPPER('pg'), UPPER('child'), TO_DATE('24/1/2020', 'dd/mm/yyyy') );
        insert into movies
            values(title_ID_SEQ.nextval, 'Black Panther: Wakanda Forever ', 'The much-anticipated sequel to 2018s Black Panther, and final film in Marvel Studios Phase 4, will be imbued with the spirit of Chadwick Boseman,
            as the story of Wakanda continues. ',
            UPPER('pg13'), UPPER('action'), TO_DATE('11/11/2022', 'dd/mm/yyyy') );
```

2.14 Movies data insertion

- Media

```
INSERT INTO media
            VALUES (media_id_seq.nextval, upper('dvd'), 1);
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('vhs'), 1);
            select * from rental_history;
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('dvd'), 2);
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('vhs'), 2);
            select * from rental_history;
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('dvd'), 3);
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('vhs'), 3);
            select * from rental_history;
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('dvd'), 4);
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('vhs'), 4);
            select * from rental_history;
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('dvd'), 5);
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('vhs'), 5);
            select * from rental_history;
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('dvd'), 6);
        INSERT INTO media
            VALUES (media_id_seq.nextval, upper('vhs'), 6);
            select * from rental_history;
```

2.15 Media data insertion

- Rental_history

```sql
INSERT INTO rental_history
            VALUES (92,'19-SEP-10', 101, '20-SEP-10' );
        INSERT INTO rental_history
            VALUES (99,default, 102, null );
        INSERT INTO rental_history
            VALUES (95,default, 104, '30-OCT-22' );
        INSERT INTO rental_history
            VALUES (101,default, 105, '16-NOV-22' );
```

2.16 Rental_history data insertion

- Actors

```sql
insert into actors
            values (actor_id_seq.nextval, 'Brad Pitt', 'William', 'Pitt', '18-DEC-1963');
        insert into actors
            values (actor_id_seq.nextval, 'Black Adam', 'Johnsonm', 'Dwayne', '02-MAY-1972');
        insert into actors
            values (actor_id_seq.nextval, 'Pennyworth', 'Bannon', 'Jack', '24-MAR-91');
        insert into actors
            values (actor_id_seq.nextval, 'Sandman', 'Jerome', 'Thomas', '21-DEC-85');
        insert into actors
            values (actor_id_seq.nextval, 'Remember the Titans', 'Gosling', 'Ryan ', '12-NOV-80');
```

2.17 Actors data insertion

- Star_billings

```sql
insert into star_billings
            values (1005, 1 ,'Action lead');

        insert into star_billings
            values (1002, 2 ,'The best fights');

        insert into star_billings
            values (1003, 3 ,'Shocking');

        insert into star_billings
            values (1004, 5 ,'Exciting');
```

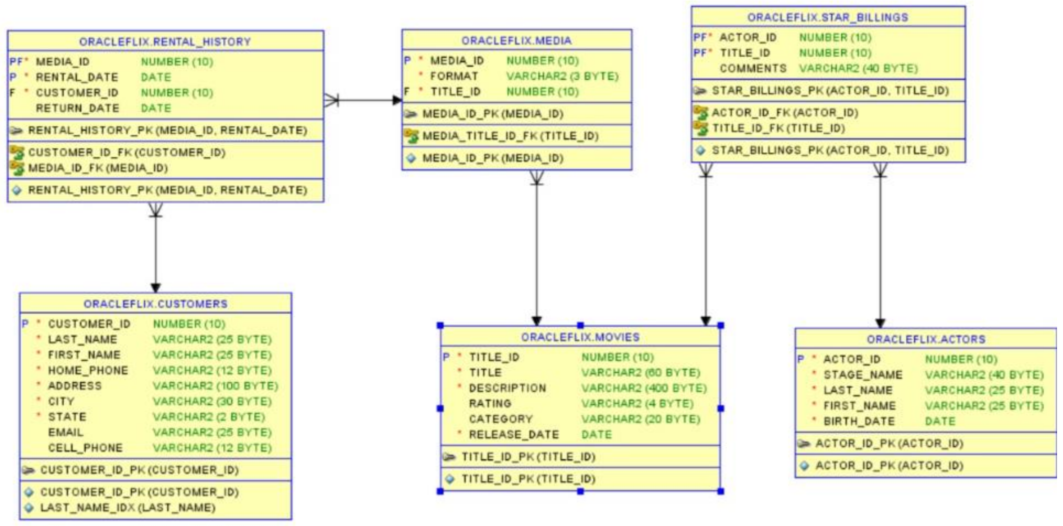
2.18 Star_billings data insertion

And the last step is to create the view Title_unavail

```sql
CREATE OR REPLACE VIEW TITLE_UNAVAIL
        AS
        SELECT M.title TITLE, s.media_id "MEDIA"
        FROM media s inner join movies M on M.title_id=s.title_id
                 inner join rental_history r on s.media_id = r.media_id
        WHERE r.return_date is null
        WITH READ ONLY;
```

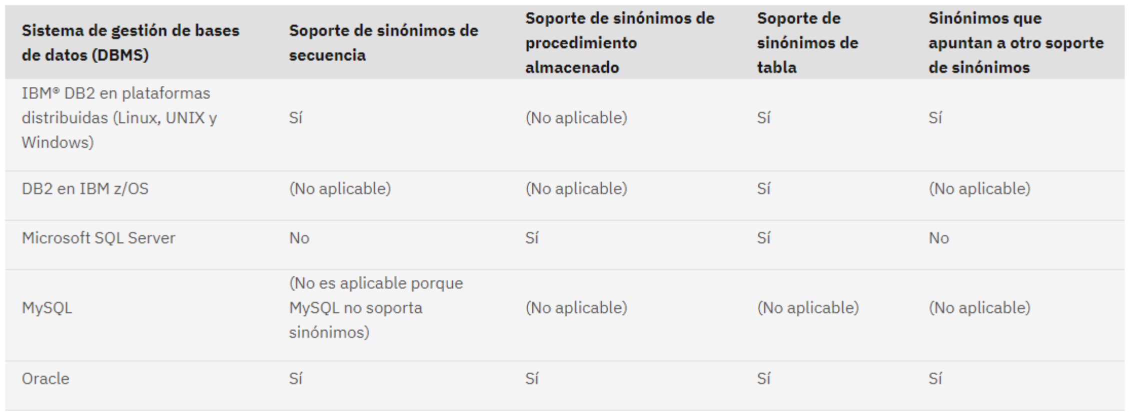
2.13 View creation

Database diagram


ORACLEFLIX.RENTAL_HISTORY
PF* MEDIA_ID NUMBER (10)
P * RENTAL_DATE DATE
F * CUSTOMER_ID NUMBER (10)
RETURN_DATE DATE
RENTAL_HISTORY_PK (MEDIA_ID, RENTAL_DATE)
CUSTOMER_ID_FK (CUSTOMER_ID)
MEDIA_ID_FK (MEDIA_ID)
RENTAL_HISTORY_PK (MEDIA_ID, RENTAL_DATE)
ORACLEFLIX.MEDIA
P * MEDIA_ID NUMBER (10)
* FORMAT VARCHAR2 (3 BYTE)
F * TITLE_ID NUMBER (10)
MEDIA_ID_PK (MEDIA_ID)
MEDIA_TITLE_ID_FK (TITLE_ID)
MEDIA_ID_PK (MEDIA_ID)
ORACLEFLIX.STAR_BILLINGS
PF* ACTOR_ID NUMBER (10)
PF* TITLE_ID NUMBER (10)
COMMENTS VARCHAR2 (40 BYTE)
STAR_BILLINGS_PK (ACTOR_ID, TITLE_ID)
ACTOR_ID_FK (ACTOR_ID)
TITLE_ID_FK (TITLE_ID)
STAR_BILLINGS_PK (ACTOR_ID, TITLE_ID)
ORACLEFLIX.CUSTOMERS
P * CUSTOMER_ID NUMBER (10)
* LAST_NAME VARCHAR2 (25 BYTE)
* FIRST_NAME VARCHAR2 (25 BYTE)
* HOME_PHONE VARCHAR2 (12 BYTE)
* ADDRESS VARCHAR2 (100 BYTE)
* CITY VARCHAR2 (30 BYTE)
* STATE VARCHAR2 (2 BYTE)
EMAIL VARCHAR2 (25 BYTE)
CELL_PHONE VARCHAR2 (12 BYTE)
CUSTOMER_ID_PK (CUSTOMER_ID)
CUSTOMER_ID_PK (CUSTOMER_ID)
LAST_NAME_IDX (LAST_NAME)
ORACLEFLIX.MOVIES
P * TITLE_ID NUMBER (10)
* TITLE VARCHAR2 (60 BYTE)
* DESCRIPTION VARCHAR2 (400 BYTE)
RATING VARCHAR2 (4 BYTE)
CATEGORY VARCHAR2 (20 BYTE)
* RELEASE_DATE DATE
TITLE_ID_PK (TITLE_ID)
TITLE_ID_PK (TITLE_ID)
ORACLEFLIX.ACTORS
P * ACTOR_ID NUMBER (10)
* STAGE_NAME VARCHAR2 (40 BYTE)
* LAST_NAME VARCHAR2 (25 BYTE)
* FIRST_NAME VARCHAR2 (25 BYTE)
* BIRTH_DATE DATE
ACTOR_ID_PK (ACTOR_ID)
ACTOR_ID_PK (ACTOR_ID)

3 ERD of the database

Reliable sources to justify that there are no synonyms in MySQL



Sistema de gestión de bases de datos (DBMS)	Soporte de sinónimos de secuencia	Soporte de sinónimos de procedimiento almacenado	Soporte de sinónimos de tabla	Sinónimos que apuntan a otro soporte de sinónimos
IBM® DB2 en plataformas distribuidas (Linux, UNIX y Windows)	Sí	(No aplicable)	Sí	Sí
DB2 en IBM z/OS	(No aplicable)	(No aplicable)	Sí	(No aplicable)
Microsoft SQL Server	No	Sí	Sí	No
MySQL	(No es aplicable porque MySQL no soporta sinónimos)	(No aplicable)	(No aplicable)	(No aplicable)
Oracle	Sí	Sí	Sí	Sí

4 Comparative table from IBM on DBMS

- https://www.ibm.com/docs/es/rtw/9.1.1?topic=overview-synonyms
- https://es.stackoverflow.com/tags/sql-server/synonyms

Conclusions:

During this project, we were challenged to design, analyzed, implement and demonstrate a real-world scenario where we implemented a database using mySQL and Oracle SQL, using as a base a conceptual representation of an organization’s information. We implemented the basic SQL syntax and the rules for constructing valid SQL statements. We made use of the general functions, conditional expressions join operations, group functions, single-row and multiple row subqueries, pair-wise and non-pair-wise subqueries, correlated subqueries, as well as DML, DDL and DCL statements. We developed freely our abilities to work as a team, and also to solve all kind of problems that could appear while we worked on the project.

Bibliography:

IBM. (March 05 2021). *Sinónimos*. Obtained from IBM: https://www.ibm.com/docs/es/rtw/9.1.1?topic=overview-synonyms

StackOverflow. (Recovered on October 10 2022). *Información de etiqueta*. Obtained from StackOverflow: https://es.stackoverflow.com/tags/sql-server/synonyms