****Технически университет - Варна

Дисциплина : Бази от данни

**ДОКУМЕНТАЦИЯ**

**Изработил:** Николай Иванов

**КСТ гр. 1а курс 3**

**Фак. № 17621301**

**СЪДЪРЖАНИЕ**

Задание…….………….………….……………….………….………….……….……….……….3

Релационен модел…………….…….………….………….………….………….………….4

Логически модел …………………….………….………….………….………….…………..5

Примерни данни ………….……….………...………….………….………….………….….6

Код за генериране на таблиците…………………….………….……………………..8

SQL команди …………….………….………………..……….………….………….…….….10

Тригери ………….………….…….……….……….………….………….………….………….13

Source code ..……………….……….…………..………….….……….………….………….15

**Курсова работа БД**

**Задание**

Да се проектира и реализира база от данни за **БИБЛИОТЕКА**, която да съхранява следната информация:

* Книга – наименование, автор, година на издаване, издателство, жанр;
* Служител – име, позиция, телефон
* Читател – име, телефон
* Заемане - книга, читател, служител, дата на заемане, срок

Правила:

* Всяка книга може да бъде заета текущо само от един читател,
* Всяка книга има един автор и едно издателство

Базата от данни трябва да е **нормализирана** и да позволява:

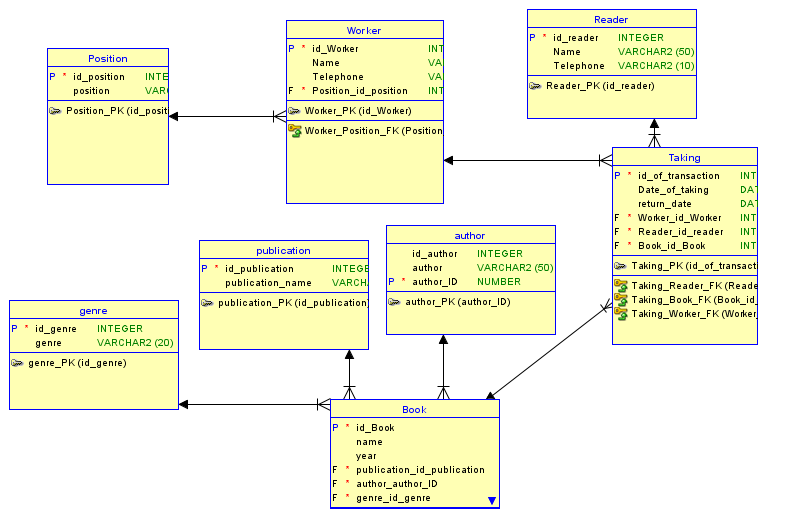
1. **Въвеждане и корекции** на данни.
2. **Търсене/заемане** на книга по: автор, ключови думи от наименованието, жанр.
3. Справки за:

* Заети книги от читател.
* Невърнати книги, подредени по дата.

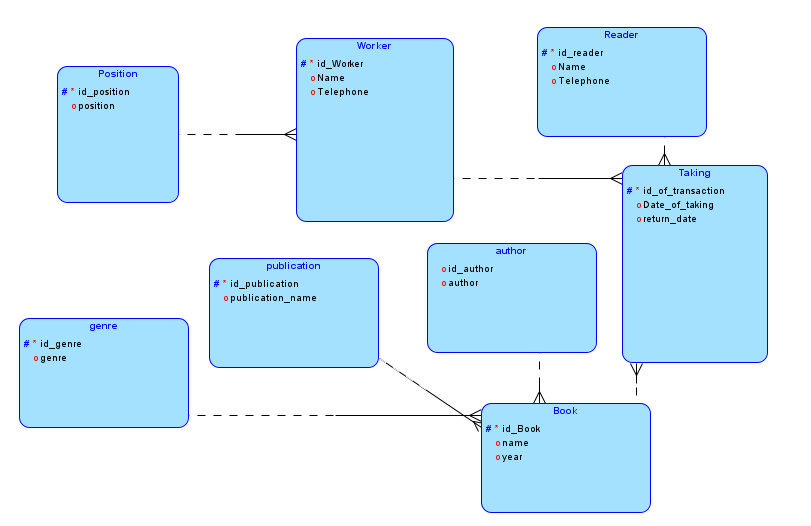
Обяснителната записка към проекта трябва да съдържа:

* Задание
* Моделите
* Примерни данни
* SQL – команди DLL, DML
* Резултати от изпълнението на заявките
* PI/SQL тригери

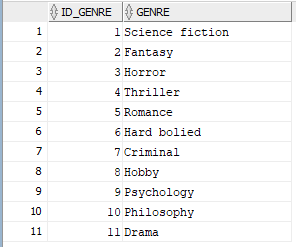
**Модели на таблиците**

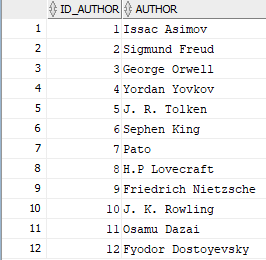
**Релационен модел:**

**Логически модел:**

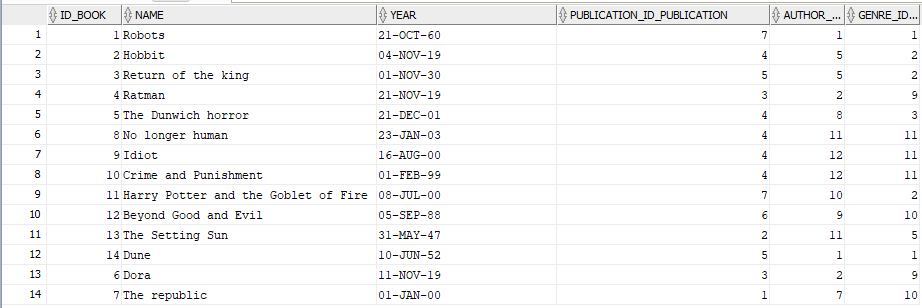


**Примерни данни:**

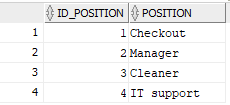
  **АВТОР: ЖАНЪР:**



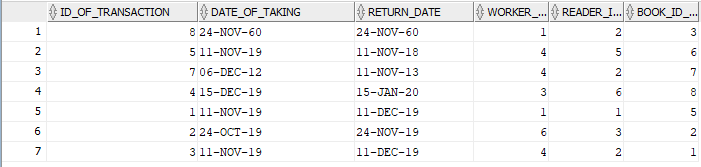
**КНИГА:**



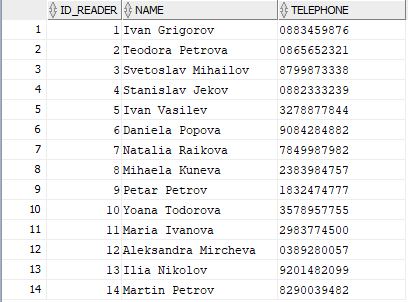
**ИЗДАТЕЛСТВО:**   **СЛУЖИТЕЛСКИ ПОЗИЦИИ:**



**ТРАНЗАНЦИЯ/ЗАЕМАНЕ:**



**ЧИТАТЕЛ/КЛИЕНТ:**



**СЛУЖИТЕЛИ:**

**Код за генериране на таблиците:**

**-- Generated by Oracle SQL Developer Data Modeler 19.2.0.182.1216**

**-- at: 2019-10-21 08:41:33 PDT**

**-- site: Oracle Database 11g**

**-- type: Oracle Database 11g**

**CREATE TABLE author (**

**id\_author INTEGER,**

**author VARCHAR2(50),**

**author\_id NUMBER NOT NULL**

**);**

**ALTER TABLE author ADD CONSTRAINT author\_pk PRIMARY KEY ( author\_id );**

**CREATE TABLE book (**

**id\_book INTEGER NOT NULL,**

**name VARCHAR2(50),**

**year DATE,**

**publication\_id\_publication INTEGER NOT NULL,**

**author\_author\_id NUMBER NOT NULL,**

**genre\_id\_genre INTEGER NOT NULL**

**);**

**ALTER TABLE book ADD CONSTRAINT book\_pk PRIMARY KEY ( id\_book );**

**CREATE TABLE genre (**

**id\_genre INTEGER NOT NULL,**

**genre VARCHAR2(20)**

**);**

**ALTER TABLE genre ADD CONSTRAINT genre\_pk PRIMARY KEY ( id\_genre );**

**CREATE TABLE position (**

**id\_position INTEGER NOT NULL,**

**position VARCHAR2(30)**

**);**

**ALTER TABLE position ADD CONSTRAINT position\_pk PRIMARY KEY ( id\_position );**

**CREATE TABLE publication (**

**id\_publication INTEGER NOT NULL,**

**publication\_name VARCHAR2(50)**

**);**

**ALTER TABLE publication ADD CONSTRAINT publication\_pk PRIMARY KEY ( id\_publication );**

**CREATE TABLE reader (**

**id\_reader INTEGER NOT NULL,**

**name VARCHAR2(50),**

**telephone VARCHAR2(10)**

**);**

**ALTER TABLE reader ADD CONSTRAINT reader\_pk PRIMARY KEY ( id\_reader );**

**CREATE TABLE taking (**

**id\_of\_transaction INTEGER NOT NULL,**

**date\_of\_taking DATE,**

**return\_date DATE,**

**worker\_id\_worker INTEGER NOT NULL,**

**reader\_id\_reader INTEGER NOT NULL,**

**book\_id\_book INTEGER NOT NULL**

**);**

**ALTER TABLE taking ADD CONSTRAINT taking\_pk PRIMARY KEY ( id\_of\_transaction );**

**CREATE TABLE worker (**

**id\_worker INTEGER NOT NULL,**

**name VARCHAR2(50),**

**telephone VARCHAR2(10),**

**position\_id\_position INTEGER NOT NULL**

**);**

**ALTER TABLE worker ADD CONSTRAINT worker\_pk PRIMARY KEY ( id\_worker );**

**ALTER TABLE book**

**ADD CONSTRAINT book\_author\_fk FOREIGN KEY ( author\_author\_id )**

**REFERENCES author ( author\_id );**

**ALTER TABLE book**

**ADD CONSTRAINT book\_genre\_fk FOREIGN KEY ( genre\_id\_genre )**

**REFERENCES genre ( id\_genre );**

**ALTER TABLE book**

**ADD CONSTRAINT book\_publication\_fk FOREIGN KEY ( publication\_id\_publication )**

**REFERENCES publication ( id\_publication );**

**ALTER TABLE taking**

**ADD CONSTRAINT taking\_book\_fk FOREIGN KEY ( book\_id\_book )**

**REFERENCES book ( id\_book );**

**ALTER TABLE taking**

**ADD CONSTRAINT taking\_reader\_fk FOREIGN KEY ( reader\_id\_reader )**

**REFERENCES reader ( id\_reader );**

**ALTER TABLE taking**

**ADD CONSTRAINT taking\_worker\_fk FOREIGN KEY ( worker\_id\_worker )**

**REFERENCES worker ( id\_worker );**

**ALTER TABLE worker**

**ADD CONSTRAINT worker\_position\_fk FOREIGN KEY ( position\_id\_position )**

**REFERENCES position ( id\_position );**

**CREATE SEQUENCE author\_author\_id\_seq START WITH 1 NOCACHE ORDER;**

**CREATE OR REPLACE TRIGGER author\_author\_id\_trg BEFORE**

**INSERT ON author**

**FOR EACH ROW**

**WHEN ( new.author\_id IS NULL )**

**BEGIN**

**:new.author\_id := author\_author\_id\_seq.nextval;**

**END;**

**/**

**КОМАНДИ ЗА ВЪВЕЖДАНЕ НА ДАННИ В ТАБЛИЦИТЕ:**

INSERT INTO book VALUES( bookseq.nextval, '&book\_name' , '&book\_date' , &book\_publication\_id ,&book\_author\_id ,&book\_genre\_id);

INSERT INTO reader VALUES(READERSEQ.nextval,'&reader\_name','&reader\_telephone');

INSERT INTO worker VALUES (WORKERSEQ.nextval, '&worker\_name' , '&worker\_telephone' , &worker\_position);

INSERT INTO publication VALUES(PUBLICATIONSEQ.nextval, '&publication');

INSERT INTO position\_ VALUES(POSITIONSEQ.nextval, '&position');

INSERT INTO genre VALUES(GENRESEQ.nextval, '&genre');

INSERT INTO author VALUES(AUTHORSEQ.nextval, '&author');

**КОМАНДИ ЗА КОРЕГИРАНЕ НА ДАННИ ОТ ТАБЛИЦИТЕ:**

UPDATE book SET name = '&name' WHERE id\_book = &selected\_id;

UPDATE reader SET name = '&name' WHERE id\_reader = &selected\_id;

UPDATE worker SET name = '&name' WHERE id\_worker = &selected\_id;

UPDATE publication SET publication\_name = '&name' WHERE id\_publication = &selected\_id;

UPDATE author SET author = '&name' WHERE id\_author = &selected\_id;

UPDATE genre SET genre = '&genre' WHERE id\_genre = &selected\_id;

UPDATE position\_ set position = '&position' WHERE id\_position = &selected\_id;

**КОМАНДИ ЗА :**

**Търсене на книга по ключова дума:**

SELECT BOOK.name , BOOK.year ,publication.publication\_name ,AUTHOR.AUTHOR ,genre.genre FROM BOOK

INNER JOIN GENRE

ON book.genre\_id\_genre = genre.id\_genre

INNER JOIN AUTHOR

ON book.AUTHOR\_AUTHOR\_ID = AUTHOR.id\_author

INNER JOIN PUBLICATION

ON book.PUBLICATION\_ID\_PUBLICATION = publication.id\_publication -- vsichko

WHERE BOOK.NAME LIKE '%&selected%';

При въведен “republic” на SQL изкарва “The republic”



**Търсене на книга по жанр:**

SELECT BOOK.name , BOOK.year ,publication.publication\_name ,AUTHOR.AUTHOR ,genre.genre FROM BOOK

INNER JOIN GENRE

ON book.genre\_id\_genre = genre.id\_genre

INNER JOIN AUTHOR

ON book.AUTHOR\_AUTHOR\_ID = AUTHOR.id\_author

INNER JOIN PUBLICATION

ON book.PUBLICATION\_ID\_PUBLICATION = publication.id\_publication

WHERE genre\_id\_genre = &Genre\_ID;

****При въведено “1” (асоцийран с sci-fi жанра) SQL изкарва:

**Търсене на книга по автор:**

SELECT BOOK.name , BOOK.year ,publication.publication\_name ,AUTHOR.AUTHOR ,genre.genre FROM BOOK

INNER JOIN GENRE

ON book.genre\_id\_genre = genre.id\_genre

INNER JOIN AUTHOR

ON book.AUTHOR\_AUTHOR\_ID = AUTHOR.id\_author

INNER JOIN PUBLICATION

ON book.PUBLICATION\_ID\_PUBLICATION = publication.id\_publication

WHERE author\_author\_id = &Author\_ID;

При въведенo “8” (асоцийран с автора H.P Lovecraft) SQL изкарва:

****

**Книги взети от читател:**

SELECT taking.book\_id\_book, book.name FROM taking

INNER JOIN book

ON taking.book\_id\_book = book.id\_book

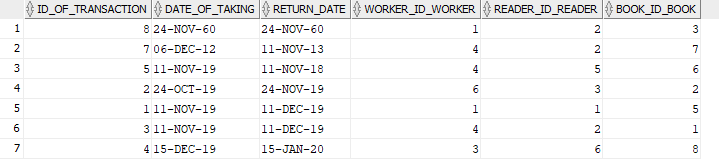
WHERE taking.reader\_id\_reader = &reader\_ID;

При въведен “1” ” (асоцийран с първи чтател) SQL изкарва:



**Невърнати книги, подредени по дата:**

SELECT \* FROM taking ORDER BY return\_date;



**ТРИГЕРИ:**

**Тригер за генериране на ID за нов работник:**

create or replace trigger WORKERTRIGGER

before insert on "HR"."WORKER"

for each row

begin

if inserting then

if :NEW."ID\_WORKER" is null then

select WORKERSEQ.nextval into :NEW."ID\_WORKER" from dual;

end if;

end if;

end;

**Тригер за генериране на ID за нов автор:**

create or replace trigger AuthorTrigger

before insert on "HR"."AUTHOR"

for each row

begin

if inserting then

if :NEW."ID\_AUTHOR" is null then

select AUTHORSEQ.nextval into :NEW."ID\_AUTHOR" from dual;

end if;

end if;

end;

**Тригер за генериране на ID за нов жанър:**

create or replace trigger genreTrigger

before insert on "HR"."GENRE"

for each row

begin

if inserting then

if :NEW."ID\_GENRE" is null then

select GENRESEQ.nextval into :NEW."ID\_GENRE" from dual;

end if;

end if;

end;

**Тригер за генериране на ID за нова позиция:**

create or replace trigger POSITIONTRIGGER

before insert on "HR"."POSITION\_"

for each row

begin

if inserting then

if :NEW."ID\_POSITION" is null then

select POSITIONSEQ.nextval into :NEW."ID\_POSITION" from dual;

end if;

end if;

end;

**Тригер за генериране на ID за нова побликация:**

create or replace trigger PUBLICATIONSEQ

before insert on "HR"."PUBLICATION"

for each row

begin

if inserting then

if :NEW."ID\_PUBLICATION" is null then

select POSITIONSEQ.nextval into :NEW."ID\_PUBLICATION" from dual;

end if;

end if;

end;

**Тригер за генериране на ID за нов читател:**

create or replace trigger READERTRIGGER

before insert on "HR"."READER"

for each row

begin

if inserting then

if :NEW."ID\_READER" is null then

select READERSEQ.nextval into :NEW."ID\_READER" from dual;

end if;

end if;

end;

**Тригер за генериране на ID за ново заемане:**

create or replace trigger TAKINGTRIGGER

before insert on "HR"."TAKING"

for each row

begin

if inserting then

if :NEW."ID\_OF\_TRANSACTION" is null then

select TAKINGSEQ.nextval into :NEW."ID\_OF\_TRANSACTION" from dual;

end if;

end if;

end;

**Тригер за генериране на ID за нова книга:**

create or replace trigger newnwnewtrigger

before insert on "HR"."BOOK"

for each row

begin

if inserting then

if :NEW."ID\_BOOK" is null then

select BOOKSEQ.nextval into :NEW."ID\_BOOK" from dual;

end if;

end if;

end;

**Source code:**

ALTER TABLE taking ADD CONSTRAINT taking\_bbq UNIQUE (book\_id\_book);

INSERT INTO book VALUES( bookseq.nextval, '&book\_name' , '&book\_date' ,&book\_publication\_id, & book\_author\_id, &book\_genre\_id);

INSERT INTO reader VALUES(READERSEQ.nextval,'&reader\_name','&reader\_telephone');

INSERT INTO worker VALUES (WORKERSEQ.nextval, '&worker\_name', '&worker\_telephone' ,&worker\_position);

INSERT INTO publication VALUES(PUBLICATIONSEQ.nextval,'&publication');

INSERT INTO position\_ VALUES(POSITIONSEQ.nextval,'&position');

INSERT INTO genre VALUES(GENRESEQ.nextval,'&genre');

INSERT INTO author VALUES(AUTHORSEQ.nextval,'&author');

INSERT INTO taking VALUES( &transation\_id, '&taking\_date' , '&return\_date' ,&worker\_id,&reader\_id,&book\_idf);

UPDATE book SET name = '&name' WHERE id\_book = &selected\_id;

UPDATE reader SET name = '&name' WHERE id\_reader = &selected\_id;

UPDATE worker SET name = '&name' WHERE id\_worker = &selected\_id;

UPDATE publication SET publication\_name = '&name' WHERE id\_publication = &selected\_id;

UPDATE author SET author = '&name' WHERE id\_author = &selected\_id;

UPDATE genre SET genre = '&genre' WHERE id\_genre = &selected\_id;

UPDATE position\_ set position = '&position' WHERE id\_position = &selected\_id;

SELECT BOOK.name , BOOK.year ,publication.publication\_name ,AUTHOR.AUTHOR ,genre.genre FROM BOOK

INNER JOIN GENRE

ON book.genre\_id\_genre = genre.id\_genre

INNER JOIN AUTHOR

ON book.AUTHOR\_AUTHOR\_ID = AUTHOR.id\_author

INNER JOIN PUBLICATION

ON book.PUBLICATION\_ID\_PUBLICATION = publication.id\_publication

WHERE BOOK.NAME LIKE '%&selected%';

SELECT BOOK.name , BOOK.year ,publication.publication\_name ,AUTHOR.AUTHOR ,genre.genre FROM BOOK

INNER JOIN GENRE

ON book.genre\_id\_genre = genre.id\_genre

INNER JOIN AUTHOR

ON book.AUTHOR\_AUTHOR\_ID = AUTHOR.id\_author

INNER JOIN PUBLICATION

ON book.PUBLICATION\_ID\_PUBLICATION = publication.id\_publication

WHERE genre\_id\_genre = &Genre\_ID;

SELECT BOOK.name , BOOK.year ,publication.publication\_name ,AUTHOR.AUTHOR ,genre.genre FROM BOOK

INNER JOIN GENRE

ON book.genre\_id\_genre = genre.id\_genre

INNER JOIN AUTHOR

ON book.AUTHOR\_AUTHOR\_ID = AUTHOR.id\_author

INNER JOIN PUBLICATION

ON book.PUBLICATION\_ID\_PUBLICATION = publication.id\_publication

WHERE author\_author\_id = &Author\_ID;

SELECT taking.book\_id\_book, book.name FROM taking

INNER JOIN book

ON taking.book\_id\_book = book.id\_book

WHERE taking.reader\_id\_reader = &reader\_ID;

SELECT \* FROM taking ORDER BY return\_date;