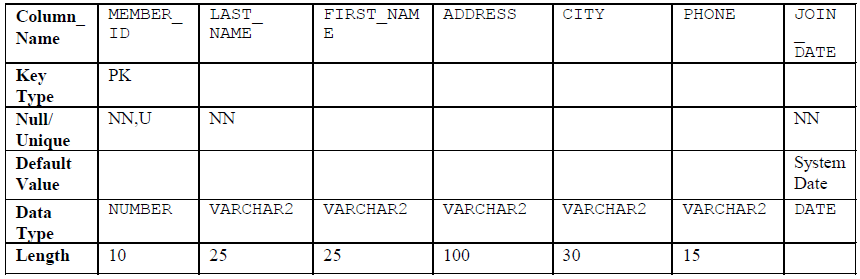
***SQL Workshop***

**Practice 14 Solutions**

1. Create the tables based on the following table instance charts. Choose the appropriate data types and be sure to add integrity constraints.

1. Table name: MEMBER



**CREATE TABLE member**

**(member\_id NUMBER(10)**

**CONSTRAINT member\_member\_id\_pk PRIMARY KEY,**

**last\_name VARCHAR2(25)**

**CONSTRAINT member\_last\_name\_nn NOT NULL,**

**first\_name VARCHAR2(25),**

**address VARCHAR2(100),**

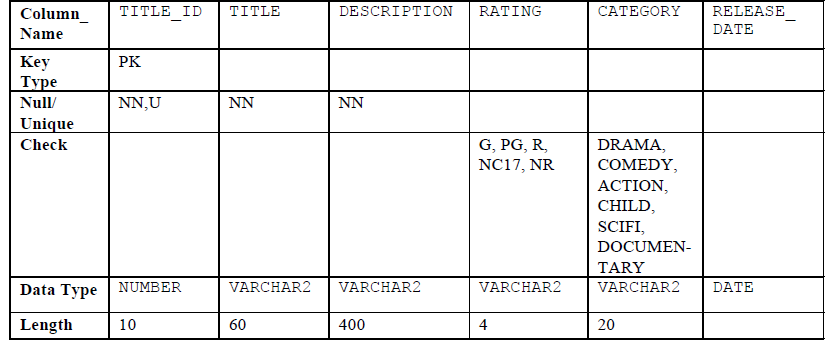
**city VARCHAR2(30),**

**phone VARCHAR2(15),**

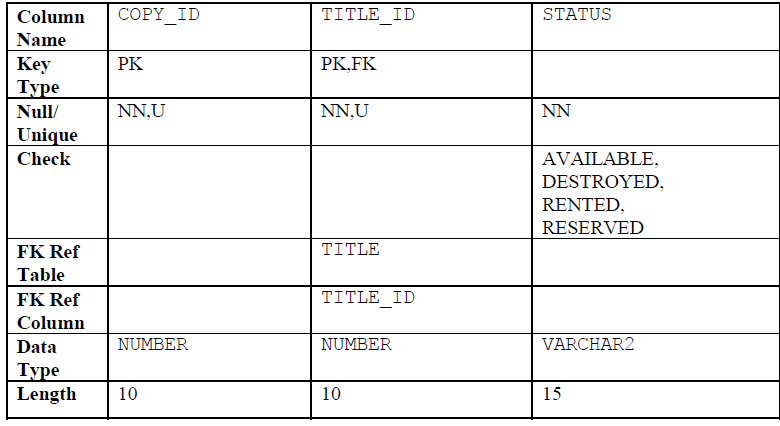
**join\_date DATE DEFAULT SYSDATE**

**CONSTRAINT member\_join\_date\_nn NOT NULL);**

1. Table name: TITLE

****

1. Table name: TITLE\_COPY

****

**CREATE TABLE title\_copy**

**(copy\_id NUMBER(10),**

**title\_id NUMBER(10)**

**CONSTRAINT title\_copy\_title\_if\_fk REFERENCES title(title\_id),**

**status VARCHAR2(15)**

**CONSTRAINT title**\_**copy**\_**status**\_**nn NOT NULL**

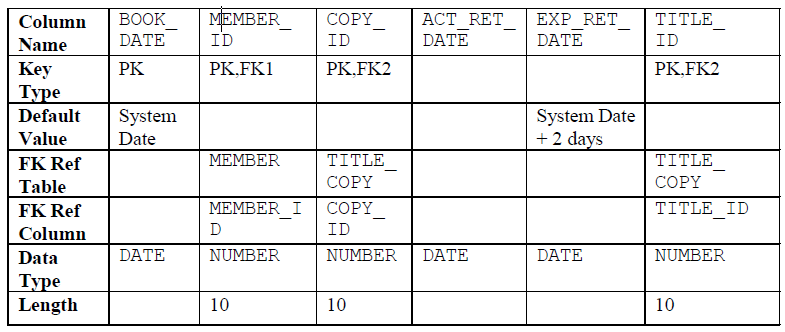
**CONSTRAINT title\_copy\_status\_ck CHECK (status IN**

**('AVAILABLE', 'DESTROYED','RENTED', 'RESERVED')),**

**CONSTRAINT title\_copy\_copy\_id\_title\_id\_pk**

**PRIMARY KEY (copy\_id, title\_id));**

1. Table name: RENTAL



**CREATE TABLE rental**

**(book\_date DATE DEFAULT SYSDATE,**

**member\_id NUMBER(10)**

**CONSTRAINT rental**\_**member**\_**id**\_**fk**

**REFERENCES member(member\_id),**

**copy\_id NUMBER(10),**

**act\_ret\_date DATE,**

**exp\_ret\_date DATE DEFAULT SYSDATE + 2,**

**title\_id NUMBER(10),**

**CONSTRAINT rental\_book\_date\_copy\_title\_pk**

**PRIMARY KEY (book\_date, member\_id,**

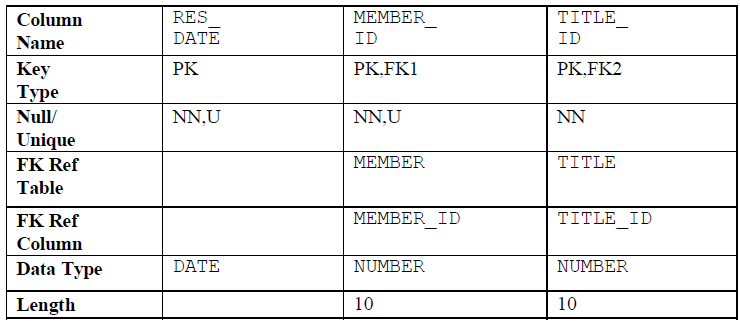
**copy\_id,title\_id),**

**CONSTRAINT rental\_copy\_id\_title\_id\_fk**

**FOREIGN KEY (copy\_id, title\_id)**

**REFERENCES title\_copy(copy\_id, title\_id));**

1. Table name: RESERVATION



**CREATE TABLE reservation**

**(res\_date DATE,**

**member\_id NUMBER(10)**

**CONSTRAINT reservation\_member\_id**

**REFERENCES member(member\_id),**

**title\_id NUMBER(10)**

**CONSTRAINT reservation\_title\_id**

**REFERENCES title(title\_id),**

**CONSTRAINT reservation\_resdate\_mem\_tit\_pk PRIMARY KEY**

**(res\_date, member\_id, title\_id));**

2. Verify that the tables and constraints were created properly by checking the data dictionary.

**SELECT table\_name**

**FROM user\_tables**

**WHERE table\_name IN ('MEMBER', 'TITLE', 'TITLE\_COPY',**

**'RENTAL', 'RESERVATION');**

**SELECT constraint\_name, constraint\_type, table\_name**

**FROM user\_constraints**

**WHERE table\_name IN ('MEMBER', 'TITLE', 'TITLE\_COPY',**

**'RENTAL', 'RESERVATION');**

3. Create sequences to uniquely identify each row in the MEMBER table and the TITLE table.

a. Member number for the MEMBER table: start with 101; do not allow caching of the

values. Name the sequence MEMBER\_ID\_SEQ.

**CREATE SEQUENCE member**\_**id**\_**seq**

**START WITH 101**

**NOCACHE;**

b. Title number for the TITLE table: start with 92; no caching. Name the sequence

TITLE\_ID\_SEQ.

**CREATE SEQUENCE title**\_**id**\_**seq**

**START WITH 92**

**NOCACHE;**

c. Verify the existence of the sequences in the data dictionary.

**SELECT sequence\_name, increment\_by, last\_number**

**FROM user\_sequences**

**WHERE sequence\_name IN ('MEMBER\_ID\_SEQ', 'TITLE\_ID\_SEQ ');**

4. Add data to the tables. Create a script for each set of data to add.

a. Add movie titles to the TITLE table. Write a script to enter the movie information. Save the

statements in a script named lab14\_4a.sql*.* Use the sequences to uniquely identify each

title. Enter the release dates in the DD-MON-YYYY format. Remember that single quotation

marks in a character field must be specially handled. Verify your additions.

**SET ECHO OFF**

**INSERT INTO title(title\_id, title, description, rating,**

**category, release\_date)**

**VALUES (title\_id\_seq.NEXTVAL, 'Willie and Christmas Too',**

**'All of Willie''s friends make a Christmas list for**

**Santa, but Willie has yet to add his own wish list.',**

**'G', 'CHILD', TO\_DATE('05-OCT-1995','DD-MON-YYYY')**

**/**

**INSERT INTO title(title**\_**id , title, description, rating,**

**category, release**\_**date)**

**VALUES (title\_id\_seq.NEXTVAL, 'Alien Again', 'Yet another**

**installment of science fiction history. Can the**

**heroine save the planet from the alien life form?',**

**'R', 'SCIFI', TO\_DATE( '19-MAY-1995','DD-MON-YYYY'))**

**/**

**INSERT INTO title(title**\_**id, title, description, rating,**

**category, release**\_**date)**

**VALUES (title\_id\_seq.NEXTVAL, 'The Glob', 'A meteor crashes**

**near a small American town and unleashes carnivorous**

**goo in this classic.', 'NR', 'SCIFI',**

**TO\_DATE( '12-AUG-1995','DD-MON-YYYY'))**

**/**

**INSERT INTO title(title\_id, title, description, rating,**

**category, release\_date)**

**VALUES (title\_id\_seq.NEXTVAL, 'My Day Off', 'With a little**

**luck and a lot ingenuity, a teenager skips school for**

**a day in New York.', 'PG', 'COMEDY',**

**TO\_DATE( '12-JUL-1995','DD-MON-YYYY'))**

**/**

**...**

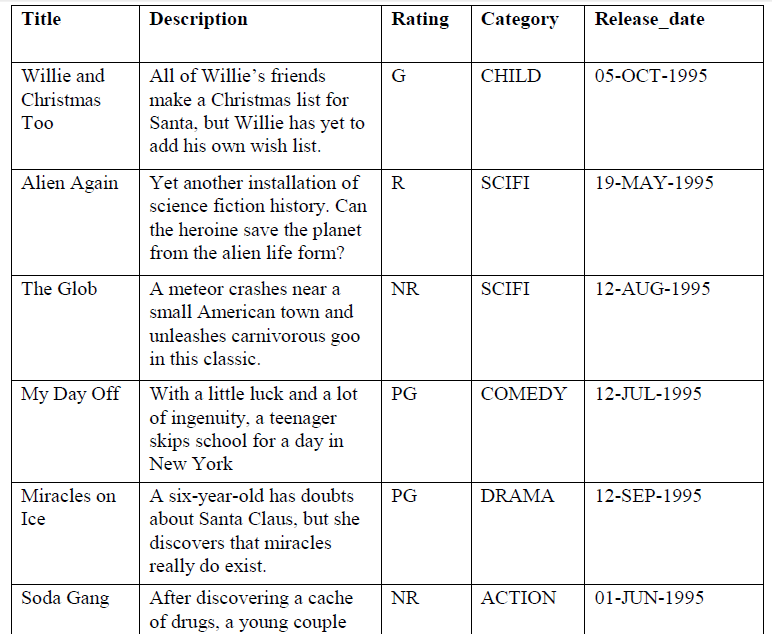
**COMMIT**

**/**

**SET ECHO ON**

**SELECT title**

**FROM title;**





b. Add data to the MEMBER table. Place the insert statements in a script named

lab14\_4b.sql. Execute commands in the script. Be sure to use the sequence to add the

member numbers.



**SET ECHO OFF**

**SET VERIFY OFF**

**INSERT INTO member(member\_id, first\_name, last\_name, address,**

**city, phone, join\_date)**

**VALUES (member\_id\_seq.NEXTVAL, '&first\_name', '&last\_name',**

**'&address', '&city', '&phone', TO\_DATE('&join\_date',**

**'DD-MM-YYYY');**

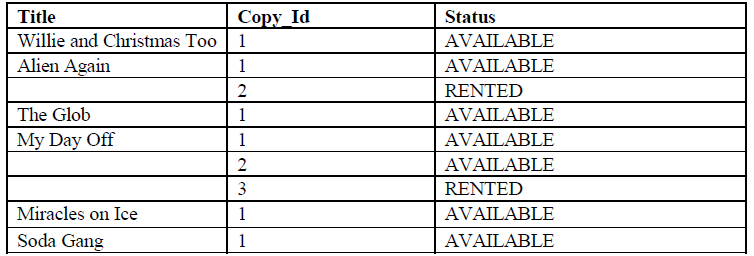
**COMMIT;**

**SET VERIFY ON**

**SET ECHO ON**

c. Add the following movie copies in the TITLE\_COPY table:

**Note:** Have the TITLE\_ID numbers available for this exercise.



**INSERT INTO title\_copy(copy\_id, title\_id, status)**

**VALUES (1, 92, 'AVAILABLE');**

**INSERT INTO title\_copy(copy\_id, title\_id, status)**

**VALUES (1, 93, 'AVAILABLE');**

**INSERT INTO title\_copy(copy\_id, title\_id, status)**

**VALUES (2, 93, 'RENTED');**

**INSERT INTO title\_copy(copy\_id, title\_id, status)**

**VALUES (1, 94, 'AVAILABLE');**

**INSERT INTO title\_copy(copy\_id, title\_id, status)**

**VALUES (1, 95, 'AVAILABLE');**

**INSERT INTO title\_copy(copy\_id, title\_id,status)**

**VALUES (2, 95, 'AVAILABLE');**

**INSERT INTO title\_copy(copy\_id, title\_id,status)**

**VALUES (3, 95, 'RENTED');**

**INSERT INTO title\_copy(copy\_id, title\_id,status)**

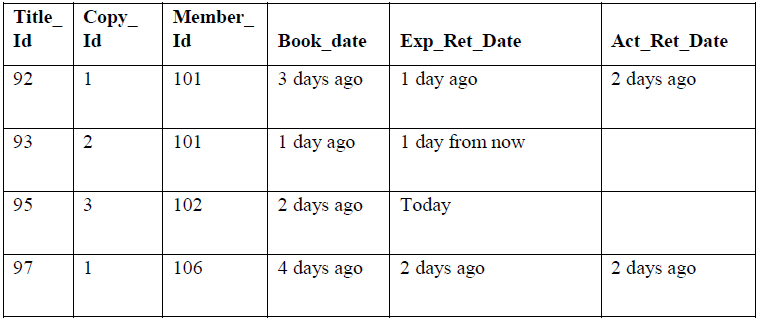
**VALUES (1, 96, 'AVAILABLE');**

**INSERT INTO title\_copy(copy\_id, title\_id,status)**

**VALUES (1, 97, 'AVAILABLE');**

d. Add the following rentals to the RENTAL table:

**Note:** Title number may be different depending on sequence number.



**INSERT INTO rental(title\_id, copy\_id, member\_id,**

**book\_date, exp\_ret\_date, act\_ret\_date)**

**VALUES (92, 1, 101, sysdate-3, sysdate-1, sysdate-2);**

**INSERT INTO rental(title\_id, copy\_id, member\_id,**

**book\_date, exp**\_**ret**\_**date, act**\_**ret**\_**date)**

**VALUES (93, 2, 101, sysdate-1, sysdate-1, NULL);**

**INSERT INTO rental(title\_id, copy\_id, member\_id,**

**book\_date, exp\_ret\_date, act\_ret\_date)**

**VALUES (95, 3, 102, sysdate-2, sysdate, NULL);**

**INSERT INTO rental(title\_id, copy\_id, member\_id,**

**book\_date, exp\_ret\_date,act\_ret\_date)**

**VALUES (97, 1, 106, sysdate-4, sysdate-2, sysdate-2);**

**COMMIT;**

5. Create a view named TITLE\_AVAIL to show the movie titles and the availability of

each copy and its expected return date if rented. Query all rows from the view. Order the results by

title.

**CREATE VIEW title\_avail AS**

**SELECT t.title, c.copy\_id, c.status, r.exp\_ret\_date**

**FROM title t, title\_copy c, rental r**

**WHERE t.title\_id = c.title\_id**

**AND c.copy\_id = r.copy\_id(+)**

**AND c.title\_id = r.title\_id(+);**

**SELECT \***

**FROM title\_avail**

**ORDER BY title, copy\_id;**

6. Make changes to data in the tables.

a. Add a new title. The movie is “Interstellar Wars,” which is rated PG and classified as a

scifi movie. The release date is 07-JUL-77. The description is “Futuristic interstellar

action movie. Can the rebels save the humans from the evil empire?” Be sure to add a title

copy record for two copies.

**INSERT INTO title(title\_id, title, description, rating,**

**category, release\_date)**

**VALUES (title\_id\_seq.NEXTVAL, 'Interstellar Wars',**

**'Futuristic interstellar action movie. Can the**

**rebels save the humans from the evil Empire?',**

**'PG', 'SCIFI', '07-JUL-77');**

**INSERT INTO title\_copy (copy\_id, title\_id, status)**

**VALUES (1, 98, 'AVAILABLE');**

**INSERT INTO title\_copy (copy\_id, title\_id, status)**

**VALUES (2, 98, 'AVAILABLE');**

b. Enter two reservations. One reservation is for Carmen Velasquez, who wants to rent

“Interstellar Wars.” The other is for Mark Quick-to-See, who wants to rent “Soda Gang.”

**INSERT INTO reservation (res\_date, member\_id, title\_id)**

**VALUES (SYSDATE, 101, 98);**

**INSERT INTO reservation (res\_date, member\_id, title\_id)**

**VALUES (SYSDATE, 104, 97);**

c. Customer Carmen Velasquez rents the movie “Interstellar Wars,” copy 1. Remove her

reservation for the movie. Record the information about the rental. Allow the default

value for the expected return date to be used. Verify that the rental was recorded by using

the view you created.

**INSERT INTO rental(title\_id, copy\_id, member\_id)**

**VALUES (98,1,101);**

**UPDATE title\_copy**

**SET status= 'RENTED'**

**WHERE title\_id = 98**

**AND copy\_id = 1;**

**DELETE**

**FROM reservation**

**WHERE member\_id = 101;**

**SELECT \***

**FROM title\_avail**

**ORDER BY title, copy\_id;**

7. Make a modification to one of the tables.

a. Add a PRICE column to the TITLE table to record the purchase price of the video. The

column should have a total length of eight digits and two decimal places. Verify your

modifications.

**ALTER TABLE title**

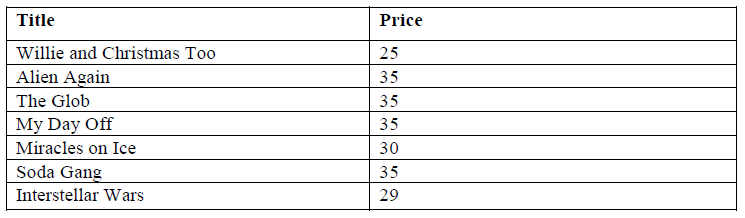
**ADD (price NUMBER(8,2));**

**DESCRIBE title;**

b. Create a script named lab14\_7b.sql that contains update statements that update each

video with a price according to the following list. Run the commands in the script.

**Note:** Have the TITLE\_ID numbers available for this exercise.

****

**SET ECHO OFF**

**SET VERIFY OFF**

**DEFINE price=**

**DEFINE title\_id=**

**UPDATE title**

**SET price = &price**

**WHERE title\_id = &title\_id;**

**SET VERIFY OFF**

**SET ECHO OFF**

c. Ensure that in the future all titles contain a price value. Verify the constraint.

**ALTER TABLE title**

**MODIFY (price CONSTRAINT title\_price\_nn NOT NULL);**

**SELECT constraint\_name, constraint\_type,**

**search**\_**condition**

**FROM user\_constraints**

**WHERE table\_name = 'TITLE';**

8. Create a report titled Customer History Report. This report contains each customer's

history of renting videos. Be sure to include the customer name, movie rented, dates of the

rental, and duration of rentals. Total the number of rentals for all customers for the reporting

period. Save the commands that generate the report in a script file named lab14\_8.sql*.*

**SET ECHO OFF**

**SET VERIFY OFF**

**TTITLE 'Customer History Report'**

**BREAK ON member SKIP 1 ON REPORT**

**SELECT m.first\_name||' '||m.last\_name MEMBER, t.title,**

**r.book\_date, r.act\_ret\_date - r.book\_date DURATION**

**FROM member m, title t, rental r**

**WHERE r.member\_id = m.member\_id**

**AND r.title\_id = t.title\_id**

**ORDER BY member;**

**CLEAR BREAK**

**TTITLE OFF**

**SET VERIFY ON**

**SET ECHO ON**