#### **DEPARTMENT OF COMPUTER SCIENCE**

#### **RAJAGIRI COLLEGE OF SOCIAL SCIENCES**

#### **(Autonomous)**

**KALAMASSERY - KOCHI - 683104**

****

### MASTER OF COMPUTER APPLICATIONS

### DBMS LAB RECORD

**NAME : MUHAMMAD ANSHAD P A**

**SEMESTER : FIRST Semester**

**REGISTER NO. : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

#### **DEPARTMENT OF COMPUTER SCIENCE**

#### **RAJAGIRI COLLEGE OF SOCIAL SCIENCES**

#### **(Autonomous)**

**KALAMASSERY - KOCHI - 683104**

****

### MASTER OF COMPUTER APPLICATIONS

### DBMS

### LAB RECORD

**NAME : MUHAMMAD ANSHAD P A**

**SEMESTER : FIRST Semester**

**REGISTER NO. : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

#### **Logo**

#### **DEPARTMENT OF COMPUTER SCIENCE**

#### **RAJAGIRI COLLEGE OF SOCIAL SCIENCES**

#### **(Autonomous)**

KALAMASSERY - KOCHI – 683104

### MASTER OF COMPUTER APPLICATIONS

###### CERTIFICATE

NAME : **MUHAMMAD ANSHAD P A**

**SEMESTER : FIRST Semester**

REGISTER NO. : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Certified that this is a bonafide record of work done by the student in the Software Laboratory of Rajagiri Department of Computer Science, Kalamassery.

Faculty in Charge Dean, Computer Science

Internal Examiner External Examiner

Place : Kalamassery

Date :

**Table of Contents**

**Activity Page No**

1. **E-R Diagram & Table Design**
2. **Practice SQL Data Definition Language(DDL) commands** 
   1. Table creation and alteration
3. **Practice SQL Data Manipulation Language (DML) commands** 
   1. Row insertion, deletion and updating
   2. Retrieval of data (Simple select query and select with

where options (include all relational and logical operators)

* 1. Functions: Numeric Data, Character Conversion and Group functions.
  2. Data manipulations using date functions
  3. Set Operations
  4. Illustration of Group by Having Clause
  5. Sub Queries

SQL Views ****

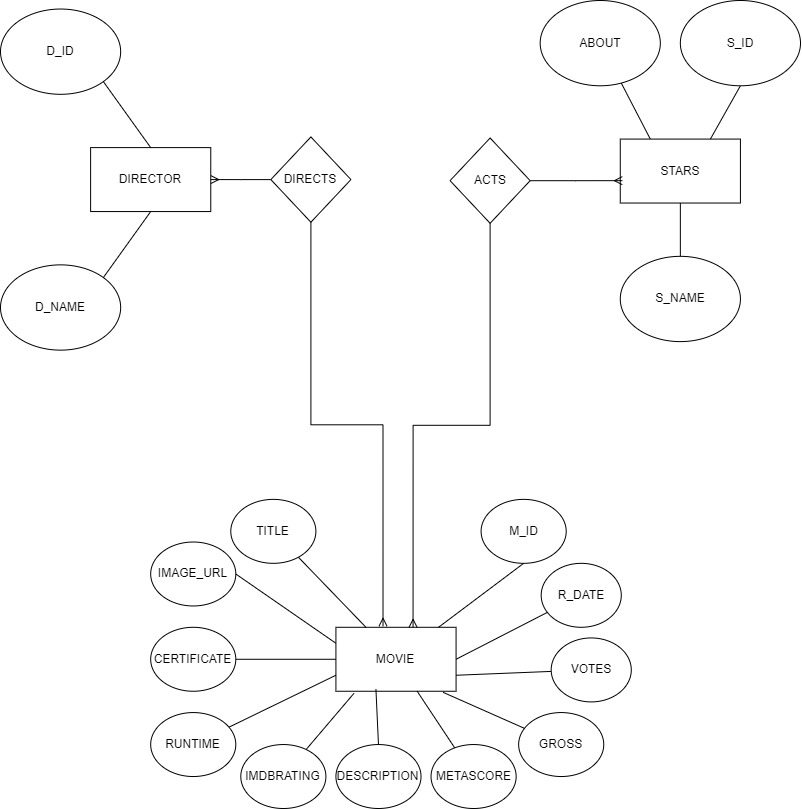
1. **Practice PL/SQL**
   1. Introductory programs
   2. Illustration of Cursors
   3. Illustration of Procedures
   4. Illustration of functions
   5. Illustration of Triggers

|  |
| --- |
| **Activity #1** |
| **E-R Diagram & Table Design** |

|  |  |
| --- | --- |
| **Description** | **Creating ER Diagrams, Table designs and Table descriptions** |
| **Date** | 14/08/2023 |

**ER Diagram & Table Design**

****



**TABLE : DIRECTORS**

|  |  |
| --- | --- |
| **D\_ID** | **D\_NAME** |
|  |  |

**TABLE : STARS**

|  |  |  |
| --- | --- | --- |
| **S\_ID** | **S\_NAME** | **ABOUT** |
|  |  |  |

****

**TABLE : MOVIES**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **M\_ID** | **TITLE** | **IMAGE\_URL** | **R\_DATE** | **CERITFICATE** | **RUNTIME** | **IMDBRATING** | **DESCRIPTION** | **METASCORE** | **GROSS** | **VOTES** |
|  |  |  |  |  |  |  |  |  |  |  |

**TABLE : MOVIESDIRECTORS**

|  |  |
| --- | --- |
| **MOVIESID** | **DIRECTORSID** |
|  |  |

**TABLE : MOVIESSTARS**

|  |  |
| --- | --- |
| **MOVIESID** | **STARSID** |
|  |  |

**TABLE DESIGN:-**

**Table name: Directors**

**Description: Used to store Directors Information**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Data Type** | **Constraints** |
| Id | Int | Primary Key/ Not Null |
| Name | Varchar2(40) | Not Null |

**Table name: Stars**

**Description: Used to store Stars Information**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Data Type** | **Constraints** |
| Id | Int | Primary Key/ Not Null |
| Name | Varchar2(40) | Unique |
| About | Varchar2(100) |  |

**Table name: Movies**

**Description: Used to store Movies Information**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Data Type** | **Constraints** |
| Id | Int | Primary Key/ Not Null |
| Title | Varchar2(40) | Not Null |
| R\_date | Date |  |
| Image\_url | Varchar2(100) |  |
| Certificate | Varchar2(20) |  |
| Runtime | Number(3,2) |  |
| ImdbRating | Number (3,1) | By default 0 |
| Description | Text(100) | By default Null |
| Metascore | Number (3,1) | By default 0 |
| Votes | Int | By default 0 |
| Gross | Number(10,2) | Gross amount should be greater than 10000 |

**Table name: MoviesDirectors**

**Description: Used to store Movie Directors Information**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Data Type** | **Constraints** |  |
| MoviesId | Int | Foreign Key references  Id of **Movies** table | Primary Key |
| DirectorsId | Int | Foreign Key references  Id of **Directors** table |

**Table name: MoviesStars**

**Description: Used to store Movie Stars Information**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Data Type** | **Constraints** |  |
| MoviesId | Int | Foreign Key references  Id of **Movies** table | Primary Key |
| StarsId | Int | Foreign Key references  Id of **Stars** table |

****

|  |
| --- |
| **Activity #2** |
| **Practice SQL Data Definition Language(DDL) commands** |

|  |  |
| --- | --- |
| **Description** | **Table creation and alterations using CREATE and ALTER commands.** |
| **Date** | 14/08/2023 |

* **Create the tables(DIRECTORS,STARS,MOVIES,MOVIESDIRECTORS,MOVIESSTARS) based on the given description.**

**//CREATING TABLE : DIRECTORS**

**Query**

****

SQL> create table directors(d\_id int,d\_name varchar2(40) not null,constraint prim\_of\_id primary key(d\_id));

Table created.

SQL> desc directors;

Name Null? Type

-------------------------- ---------------------- ----------------------------

D\_ID NOT NULL NUMBER(38)

D\_NAME NOT NULL VARCHAR2(40)

SQL>

SQL> select constraint\_name,constraint\_type from user\_constraints where table\_name='DIRECTORS';

CONSTRAINT\_NAME C

------------------------------ ----

SYS\_C0011410 C

PRIM\_OF\_ID P

**//CREATING TABLE :STARS**

**Query**

SQL> create table stars(s\_id int,s\_name varchar2(40) unique,about varchar2(100),constraint prime\_sid primary key(s\_id));

Table created.

SQL> desc stars;

Name Null? Type

----------------------------------------- -------- ----------------------------

S\_ID NOT NULL NUMBER(38)

S\_NAME VARCHAR2(40)

ABOUT VARCHAR2(100)

SQL> select constraint\_name,constraint\_type from user\_constraints where table\_name='STARS';

CONSTRAINT\_NAME C

--------------------------- ----

PRIME\_SID P

SYS\_C0011413 U

**//CREATING TABLE : MOVIES**

**Query**

SQL> create table movies(m\_id int,title varchar2(40) not null,r\_date date,image\_url varchar2(100),certificate varchar2(20),runtime number(3,2),imdbrating number(3,1) default(0),description varchar2(100) default(null),metascore number(3,1) default(0),votes int default(0),gross number(10,2),constraint gross\_check check(gross>10000),constraint prime\_mid primary key(m\_id));

Table created.

SQL> desc movies;

Name Null? Type

----------------------------------------- -------- ----------------------------

M\_ID NOT NULL NUMBER(38)

TITLE NOT NULL VARCHAR2(40)

R\_DATE DATE

IMAGE\_URL VARCHAR2(100)

CERTIFICATE VARCHAR2(20)

RUNTIME NUMBER(3,2)

IMDBRATING NUMBER(3,1)

DESCRIPTION VARCHAR2(100)

METASCORE NUMBER(3,1)

VOTES NUMBER(38)

GROSS NUMBER(10,2)

SQL> select constraint\_name,constraint\_type from user\_constraints where table\_name='MOVIES';

CONSTRAINT\_NAME C

--------------------------- ----

SYS\_C0011414 C

GROSS\_CHECK C

PRIME\_MID P

**//CREATING TABLE : MOVIESDIRECTORS**

**Query**

SQL> create table moviesdirectors(moviesid int,directorsid int,foreign key(moviesid) references movies(m\_id),foreign key(directorsid) references directors(d\_id),primary key(moviesid,directorsid));

Table created.

SQL> desc moviesdirectors;

Name Null? Type

----------------------------------------- -------- ----------------------------

MOVIESID NOT NULL NUMBER(38)

DIRECTORSID NOT NULL NUMBER(38)

SQL> select constraint\_name,constraint\_type from user\_constraints where table\_name='MOVIESDIRECTORS';

CONSTRAINT\_NAME C

--------------------------- ----

SYS\_C0011417 P

SYS\_C0011418 R****

SYS\_C0011419 R

**//CREATING TABLE : MOVIESSTARS**

**Query**

SQL> create table moviesstars(moviesid int,starsid int,foreign key(moviesid) references movies(m\_id),foreign key(starsid) references stars(s\_id),primary key(moviesid,starsid));

Table created.

SQL> desc moviesstars;

Name Null? Type

--------------------------------- --------- ---------------------------

MOVIESID NOT NULL NUMBER(38)

STARSID NOT NULL NUMBER(38)

SQL> select constraint\_name,constraint\_type from user\_constraints where table\_name='MOVIESSTARS';

CONSTRAINT\_NAME C

---------------------------- ---

SYS\_C0011420 P

SYS\_C0011421 R

SYS\_C0011422 R

* **Add a column ‘DOB’ to Stars table.**

**Query**

SQL> alter table stars add dob date;

Table altered.

SQL> desc stars;

Name Null? Type

----------------------------------------- -------- ----------------------------

S\_ID NOT NULL NUMBER(38)

S\_NAME VARCHAR2(40)

ABOUT VARCHAR2(100)

DOB DATE

* **Drop the column ‘Gross’ in Movies table.**

**Query**

SQL> alter table movies drop column gross;

Table altered.

****

SQL> desc movies;

Name Null? Type

----------------------------------------- -------- ----------------------------

M\_ID NOT NULL NUMBER(38)

TITLE NOT NULL VARCHAR2(40)

R\_DATE DATE

IMAGE\_URL VARCHAR2(100)

CERTIFICATE VARCHAR2(20)

RUNTIME NUMBER(3,2)

IMDBRATING NUMBER(3,1)

DESCRIPTION VARCHAR2(100)

METASCORE NUMBER(3,1)

VOTES NUMBER(38)

* **Add column ‘Language’ in Movies table.**

**Query**

SQL> alter table movies add language varchar2(20);

Table altered.

SQL> desc movies;

Name Null? Type

----------------------------------------- -------- ----------------------------

M\_ID NOT NULL NUMBER(38)

TITLE NOT NULL VARCHAR2(40)

R\_DATE DATE

IMAGE\_URL VARCHAR2(100)

CERTIFICATE VARCHAR2(20)

RUNTIME NUMBER(3,2)

IMDBRATING NUMBER(3,1)

DESCRIPTION VARCHAR2(100)

METASCORE NUMBER(3,1)

VOTES NUMBER(38)

LANGUAGE VARCHAR2(20)

* **Add column Gross Number(10,2) in Movies table.**

**Query**

SQL> alter table movies add gross number(12,2);

Table altered.

SQL> desc movies;

Name Null? Type

----------------------------------------- -------- ----------------------------

M\_ID NOT NULL NUMBER(38)

TITLE NOT NULL VARCHAR2(40)

R\_DATE DATE

IMAGE\_URL VARCHAR2(100)

CERTIFICATE VARCHAR2(20)

RUNTIME NUMBER(3,2)

IMDBRATING NUMBER(3,1)

DESCRIPTION VARCHAR2(100)

METASCORE NUMBER(3,1)

VOTES NUMBER(38)

LANGUAGE VARCHAR2(20)

GROSS NUMBER(12,2)

****

* **Change the name of the column ‘R\_date’ in Movies table to Releasedate.**

**Query**

SQL> alter table movies rename column r\_date to releasedate;

Table altered.

SQL> desc movies;

Name Null? Type

----------------------------------------- -------- ----------------------------

M\_ID NOT NULL NUMBER(38)

TITLE NOT NULL VARCHAR2(40)

RELEASEDATE DATE

IMAGE\_URL VARCHAR2(100)

CERTIFICATE VARCHAR2(20)

RUNTIME NUMBER(3,2)

IMDBRATING NUMBER(3,1)

DESCRIPTION VARCHAR2(100)

METASCORE NUMBER(3,1)

VOTES NUMBER(38)

LANGUAGE VARCHAR2(20)

GROSS NUMBER(12,2)

* **Add a column ‘Age’ in Directors table as Number. Age must be 7 years or above.**

**Query**

SQL> alter table directors add age int;

Table altered.

SQL> alter table directors add constraint age\_chk check(age >= 7);

Table altered.

SQL> desc directors;

Name Null? Type

----------------------------------------- -------- ----------------------------

D\_ID NOT NULL NUMBER(38)

D\_NAME NOT NULL VARCHAR2(40)

AGE NUMBER(38)

SQL> select constraint\_name,constraint\_type from user\_constraints where table\_name='DIRECTORS';

****

CONSTRAINT\_NAME C

---------------------------- ----

SYS\_C0011410 C

PRIM\_OF\_ID P

AGE\_CHK C

* **Add a new column ‘Hit’ in Movies table with datatype Number(1) and by default 0.**

**Query**

SQL> alter table movies add hit number(1) default 0;

Table altered.

SQL> desc movies;

Name Null? Type

----------------------------------------- -------- ----------------------------

M\_ID NOT NULL NUMBER(38)

TITLE NOT NULL VARCHAR2(40)

RELEASEDATE DATE

IMAGE\_URL VARCHAR2(100)

CERTIFICATE VARCHAR2(20)

RUNTIME NUMBER(3,2)

IMDBRATING NUMBER(3,1)

DESCRIPTION VARCHAR2(100)

METASCORE NUMBER(3,1)

VOTES NUMBER(38)

LANGUAGE VARCHAR2(20)

GROSS NUMBER(12,2)

HIT NUMBER(1)

* **Add a new column ‘Entry\_date’ in Movies table to record the date on which the movie details are entered in the data base.**

**Query**

SQL> alter table movies add entry\_date date;

Table altered.

SQL> desc movies;****

Name Null? Type

----------------------------------------- -------- ----------------------------

M\_ID NOT NULL NUMBER(38)

TITLE NOT NULL VARCHAR2(40)

RELEASEDATE DATE

IMAGE\_URL VARCHAR2(100)

CERTIFICATE VARCHAR2(20)

RUNTIME NUMBER(3,2)

IMDBRATING NUMBER(3,1)

DESCRIPTION VARCHAR2(100)

METASCORE NUMBER(3,1)

VOTES NUMBER(38)

LANGUAGE VARCHAR2(20)

GROSS NUMBER(12,2)

HIT NUMBER(1)

ENTRY\_DATE DATE

* **Destroy the table MoviesStars and recreate it.**

**Query**

SQL> drop table moviesstars;

Table dropped.

SQL> create table moviesstars(moviesid int,starsid int,foreign key(moviesid) references movies(m\_id),foreign key(starsid) references stars(s\_id),primary key(moviesid,starsid));

Table created.

SQL> desc moviesstars;

Name Null? Type

----------------------------------------- -------- ----------------------------

MOVIESID NOT NULL NUMBER(38)

STARSID NOT NULL NUMBER(38)

* **Change the size of the Director’s name to 30.**

**Query**

SQL> alter table directors modify d\_name varchar2(30);

Table altered.

SQL> desc directors;

Name Null? Type

----------------------------------------- -------- ----------------------------

D\_ID NOT NULL NUMBER(38)

D\_NAME NOT NULL VARCHAR2(30)

AGE NUMBER(38)

* **Add the following check constraints:**
  + **Releasedate should be less than the Entry\_date in the Movies table.**
  + **Language of movies should be Malayalam, English, Tamil or Hindi.**

**Query**

SQL> alter table movies add constraint chk\_entry\_date check(releasedate<entry\_date);

Table altered.

SQL> alter table movies add constraint chk\_language check(language in('Malayalam','English','Tamil','Hindi'));

Table altered.

SQL> select constraint\_name,constraint\_type from user\_constraints where table\_name='MOVIES';

CONSTRAINT\_NAME C

--------------------------- ----

SYS\_C0011414 C

PRIME\_MID P

CHK\_ENTRY\_DATE C

CHK\_LANGUAGE C

|  |
| --- |
| **Activity #3** |
| **Practice SQL Data Manipulation Language (DML) commands** |

|  |  |
| --- | --- |
| **Description 3.1** | **Illustration of Row insertion, deletion and updating** |
| **Date** | 14/08/2023 |

* **Insert the appropriate data (10 rows) for the tables with respect to defined datatypes, size and constraints.**

**//INSERTING VALUES TO DIRECTORS :**

**Query**

SQL> desc directors;

Name Null? Type

----------------------------------------- -------- ----------------------------

D\_ID NOT NULL NUMBER(38)

D\_NAME NOT NULL VARCHAR2(30)

AGE NUMBER(38)

****

SQL> insert into directors values('101','LAL JOSE',57);

1 row created.

SQL> insert into directors values('102','VINEETH SREENIVASAN',38);

1 row created.

SQL> insert into directors values('103','ANJALI MENON',44);

1 row created.

SQL> insert into directors values('104','S SANKAR',60);

1 row created.

SQL> insert into directors values('105','LOKESH KANAGARAJ',37);

1 row created.

SQL> insert into directors values('106','MANI RATNAM',67);

1 row created.

SQL> insert into directors values('107','RAJKUMAR HIRANI',60);

1 row created.

SQL> insert into directors values('108','NITESH TIWARI',51);

1 row created.

SQL> insert into directors values('109','JAMES CAMERON',69);

1 row created.

SQL> insert into directors values('110','CHRISTOPHER NOLAN',53);

1 row created.

SQL> select \* from directors;

D\_ID D\_NAME AGE

---------- ------------------------------ ----------

101 LAL JOSE 57

102 VINEETH SREENIVASAN 38

103 ANJALI MENON 44

104 S SANKAR 60

105 LOKESH KANAGARAJ 37

106 MANI RATNAM 67

107 RAJKUMAR HIRANI 60

108 NITESH TIWARI 51

109 JAMES CAMERON 69

110 CHRISTOPHER NOLAN 53

10 rows selected.

****

**//INSERTING VALUES TO STARS :**

**Query**

SQL> desc stars;

Name Null? Type

----------------------------------------------------- -------- ------------------------------------

S\_ID NOT NULL NUMBER(38)

S\_NAME VARCHAR2(40)

ABOUT VARCHAR2(100)

DOB DATE

SQL>

SQL> insert into stars values(501,'PRANAV MOHANLAL','MALAYALAM ACTOR','13/jul/1990');

1 row created.

SQL> insert into stars values(502,'DULQUER SALMAAN','MALAYALAM ACTOR','28/jul/1986');

1 row created.

SQL> insert into stars values(503,'DILEEP','MALAYALAM ACTOR','27/oct/1967');

1 row created.

SQL> insert into stars values(504,'RAJINIKANTH','TAMIL ACTOR','12/dec/1950');

1 row created.

SQL> insert into stars values(505,'VIJAY','TAMIL ACTOR','22/jun/1974');

1 row created.****

SQL> insert into stars values(506,'AISHWARYA RAI BACHCHAN','TAMIL ACTRESS','01/nov/1973');

1 row created.

SQL> insert into stars values(507,'AAMIR KHAN','BOLLYWOOD ACTOR','14/mar/1965');

1 row created.

SQL> insert into stars values(508,'SUSHANT SINGH RAJPUT','BOLLYWOOD ACTOR','21/jan/1986');

1 row created.

SQL> insert into stars values(509,'CILLIAN MURPHY','HOLLYWOOD ACTOR','25/may/1976');

1 row created.

SQL> insert into stars values(510,'ARNOLD SCHWARZENEGGER','HOLLYWOOD ACTOR','30/jul/1947');

1 row created.

SQL> insert into stars values(511,'ZOE SALDANA','HOLLYWOOD ACTRESS','19/jun/1979');

1 row created.

SQL> insert into stars values(512,'MATTHEW MCCONAUGHEY','HOLLYWOOD ACTOR','4/nov/1969');

1 row created.

SQL> insert into stars values(513,'PARVATHY THIRUVOTHU','MALAYALAM ACTRESS','7/apr/1988');

1 row created.

SQL> select \* from stars;

S\_ID S\_NAME ABOUT DOB

---------- ---------------------------------------- ---------------- ---------

501 PRANAV MOHANLAL MALAYALAM ACTOR 13-JUL-90

502 DULQUER SALMAAN MALAYALAM ACTOR 28-JUL-86

503 DILEEP MALAYALAM ACTOR 27-OCT-67

504 RAJINIKANTH TAMIL ACTOR 12-DEC-50

505 VIJAY TAMIL ACTOR 22-JUN-74

506 AISHWARYA RAI BACHCHAN TAMIL ACTRESS 01-NOV-73

507 AAMIR KHAN BOLLYWOOD ACTOR 14-MAR-65

508 SUSHANT SINGH RAJPUT BOLLYWOOD ACTOR 21-JAN-86

509 CILLIAN MURPHY HOLLYWOOD ACTOR 25-MAY-76

510 ARNOLD SCHWARZENEGGER HOLLYWOOD ACTOR 30-JUL-47

511 ZOE SALDANA HOLLYWOOD ACTRESS 19-JUN-79

S\_ID S\_NAME ABOUT DOB

---------- ---------------------------------------- ------------------ ---------

512 MATTHEW MCCONAUGHEY HOLLYWOOD ACTOR 04-NOV-69

513 PARVATHY THIRUVOTHU MALAYALAM ACTRESS 07-APR-88

13 rows selected.

**//INSERTING VALUES TO MOVIES :**

**Query**

SQL> desc movies;

Name Null? Type

----------------------------------------------------- -------- ------------------------------------

M\_ID NOT NULL NUMBER(38)

TITLE NOT NULL VARCHAR2(40)

RELEASEDATE DATE

IMAGE\_URL VARCHAR2(100)

CERTIFICATE VARCHAR2(20)

RUNTIME NUMBER(3,2)

IMDBRATING NUMBER(3,1)

DESCRIPTION VARCHAR2(100)

METASCORE NUMBER(3,1)

VOTES NUMBER(38)

LANGUAGE VARCHAR2(20)

GROSS NUMBER(12,2)

HIT NUMBER(1)

ENTRY\_DATE DATE

SQL>

SQL> insert into movies values(1001,'Hridayam','16/jun/2020','https://www.movies.com/Hridayam.jpg','U/A',2.34,8.4,'The emotional journery of Arun',90,93,'Malayalam',1600000000,1,'28/aug/2023');

1 row created.

SQL> insert into movies values(1002,'Meesa Madhavan','20/aug/2002','https://www.movies.com/Meesamadhavan.jpg','U',2.45,8,'Story of madhavan who is forced into a thief',92,94,'Malayalam',190000000,1,'28/aug/2023');

1 row created.

SQL> insert into movies values(1003,'Wonder women','18/nov/2022','https://www.movies.com/wonderwomen.jpg','U/A',1.2,5.2,'story of six pregnant women',60,66,'Malayalam',50000000,0,'28/aug/2023');

1 row created.

SQL> insert into movies values(1004,'Enthiran','1/oct/2010','https://www.movies.com/enthiran.jpg','U/A',2.5,7.1,'Story of humanoid robot',70,78,'Tamil',3750000000,1,'28/aug/2023');

1 row created.

SQL> insert into movies values(1005,'Master','13/jan/2021','https://www.movies.com/master.jpg','U/A',2.59,7.3,'A professor clashes with a gangster',80,87,'Tamil',2200000000,1,'28/aug/2023');

1 row created.

SQL> insert into movies values(1006,'Ponniyin Selvan:1','30/sep/2022','https://www.movies.com/ponniyinselvan1.jpg','U/A',2.5,7.6,'Chola Raja story',80,86,'Tamil',3500000000,1,'28/aug/2023');

1 row created.****

SQL> insert into movies values(1007,'3 idiots','25/dec/2009','https://www.movies.com/3idiots.jpg','U/A',2.51,8.4,'Story of 3 friends',90,94,'Hindi',4600000000,1,'28/aug/2023');

1 row created.

SQL> insert into movies values(1008,'Chichchore','6/sep/2019','https://www.movies.com/chichchore.jpg','U/A',2.23,8.3,'life of college friends',90,91,'Hindi',1820000000,1,'28/aug/2023');

1 row created.

SQL> insert into movies values(1009,'Avatar','18/dec/2009','https://www.movies.com/avatar.jpg','U/A',2.42,7.9,'Sci-fi epic',80,86,'English',2930000000,1,'28/aug/2023');

1 row created.

SQL> insert into movies values(1010,'Interstellar','7/nov/2014','https://www.movies.com/interstellar.jpg','U/A',2.49,8.7,'Ex-NASA pilot tasked to find new planet for humans',90,92,'English',7150000000,1,'28/aug/2023');

1 row created.

SQL> commit;

Commit complete.

SQL> select \* from movies;

M\_ID TITLE RELEASEDA IMAGE\_URL CERTIFICATE RUNTIME IMDBRATING DESCRIPTION

METASCORE VOTES LANGUAGE GROSS HIT ENTRY\_DAT

---------- ---------------------------------------- --------- ---------------------------------------------------------------------------------------------------- -------------------- ---------- ---------- ---------------------------------------------------------------------------------------------------- ---------- ---------- -------------------- ---------- ---------- ---------

1001 Hridayam 16-JUN-20 https://www.movies.com/Hridayam.jpg U/A 2.34 8.4 The emotional journery of Arun

90 93 Malayalam 1600000000 1 28-AUG-23

1002 Meesa Madhavan 20-AUG-02 https://www.movies.com/Meesamadhavan.jpg U 2.45 8 Story of madhavan who is forced into a thief

92 94 Malayalam 190000000 1 28-AUG-23

1003 Wonder women 18-NOV-22 https://www.movies.com/wonderwomen.jpg U/A 1.2 5.2 story of six pregnant women

60 66 Malayalam 50000000 0 28-AUG-23

1004 Enthiran 01-OCT-10 https://www.movies.com/enthiran.jpg U/A 2.5 7.1 Story of humanoid robot

70 78 Tamil 3750000000 1 28-AUG-23

1005 Master 13-JAN-21 https://www.movies.com/master.jpg U/A 2.59 7.3 A professor clashes with a gangster

80 87 Tamil 2200000000 1 28-AUG-23

1006 Ponniyin Selvan:1 30-SEP-22 https://www.movies.com/ponniyinselvan1.jpg U/A 2.5 7.6 Chola Raja story

80 86 Tamil 3500000000 1 28-AUG-23

1007 3 idiots 25-DEC-09 https://www.movies.com/3idiots.jpg U/A 2.51 8.4 Story of 3 friends

90 94 Hindi 4600000000 1 28-AUG-23

1008 Chichchore 06-SEP-19 https://www.movies.com/chichchore.jpg U/A 2.23 8.3 life of college friends

90 91 Hindi 1820000000 1 28-AUG-23

1009 Avatar 18-DEC-09 https://www.movies.com/avatar.jpg U/A 2.42 7.9 Sci-fi epic

80 86 English 2930000000 1 28-AUG-23

1010 Interstellar 07-NOV-14 https://www.movies.com/interstellar.jpg U/A 2.49 8.7 Ex-NASA pilot tasked to find new planet for humans

90 92 English 7150000000 1 28-AUG-23

10 rows selected.

****

**//INSERTING VALUES TO MOVIESDIRECTORS :**

**Query**

SQL> desc moviesdirectors;

Name Null? Type

----------------------------------------------------- -------- ------------------------------------

MOVIESID NOT NULL NUMBER(38)

DIRECTORSID NOT NULL NUMBER(38)

SQL>

SQL> insert into moviesdirectors values(1001,102);

1 row created.

SQL> insert into moviesdirectors values(1002,101);

1 row created.

SQL> insert into moviesdirectors values(1003,103);

1 row created.

SQL> insert into moviesdirectors values(1004,104);

1 row created.

SQL> insert into moviesdirectors values(1005,105);

1 row created.

SQL> insert into moviesdirectors values(1006,106);

1 row created.

SQL> insert into moviesdirectors values(1007,107);

1 row created.****

SQL> insert into moviesdirectors values(1008,108);

1 row created.

SQL> insert into moviesdirectors values(1009,109);

1 row created.

SQL> insert into moviesdirectors values(1010,110);

1 row created.

SQL> select \* from moviesdirectors;

MOVIESID DIRECTORSID

---------- -----------

1001 102

1002 101

1003 103

1004 104

1005 105

1006 106

1007 107

1008 108

1009 109

1010 110

10 rows selected.

**//INSERTING VALUES TO MOVIESSTARS :**

**Query**

SQL> desc moviesstars;

Name Null? Type

----------------------------------------------------- -------- ------------------------------------

MOVIESID NOT NULL NUMBER(38)

STARSID NOT NULL NUMBER(38)

SQL>

SQL> insert into moviesstars values(1001,501);

1 row created.

SQL> insert into moviesstars values(1002,503);

1 row created.

SQL> insert into moviesstars values(1003,513);

1 row created.

SQL> insert into moviesstars values(1004,504);

1 row created.

SQL> insert into moviesstars values(1005,505);

1 row created.

SQL> insert into moviesstars values(1006,506);

1 row created.

SQL> insert into moviesstars values(1007,507);

1 row created.

SQL> insert into moviesstars values(1008,508);

1 row created.

SQL> insert into moviesstars values(1009,511);

1 row created.

SQL> insert into moviesstars values(1010,512);

1 row created.

SQL> select \* from moviesstars;

MOVIESID STARSID

---------- ----------

1001 501

1002 503

1003 513

1004 504

1005 505

1006 506

1007 507****

1008 508

1009 511

1010 512

10 rows selected.

* **Change value of Hit to 1 where ‘Votes’ greater than or equal to 90.**

**Query**

SQL> update movies set hit=1 where (votes >= 90);

5 rows updated.

* **Create table IndustryHit with the following columns:**

**Id**

**Title**

**Releasedate**

**Language**

**Votes**

**Gross**

**The data types and null characteristics for these columns should be**

**the same as the corresponding columns in the Movies table**

**described at the beginning of the lab exercise.**

**Query**

SQL> create table industryhit(i\_id number(38),i\_title varchar2(38),i\_releasedate varchar2(40),i\_language varchar2(10),i\_votes number(38),i\_gross number(12,2),constraint prmky\_iid primary key(i\_id));

Table created.

SQL> desc industryhit;

Name Null? Type

----------------------------------------------------- -------- ------------------------------------

I\_ID NOT NULL NUMBER(38)

I\_TITLE VARCHAR2(38)

I\_RELEASEDATE VARCHAR2(40)

I\_LANGUAGE VARCHAR2(10)

I\_VOTES NUMBER(38)

I\_GROSS NUMBER(12,2)

* **New movies hit the box office; their data is as follows:**

**Id: 1014, 1021, 1032**

**Title: 2018: Everyone is a Hero, Oppenheimer, Maamannan**

**Releasedate: 5 May 2023, 21 July 2023, 29 June 2023**

**Language: Malayalam, English, Tamil**

**Votes: 97, 96, 95**

**Gross: 750000000, 500000000, 505000000**

**Add the new employees to the IndustryHit table.**

* **Insert data into the new IndustryHit table.**

**Query**

SQL> insert into industryhit values(1014,'2018:Everyone is a Hero','5/may/2023','Malayalam',97,750000000);

1 row created.

SQL> insert into industryhit values(1021,'Oppenheimer','21/jul/2023','English',96,500000000);

1 row created.

SQL> insert into industryhit values(1032,'Maamannan','29/jun/2023','Tamil',95,505000000);

1 row created.

SQL> select \* from industryhit;

I\_ID I\_TITLE I\_RELEASEDATE I\_LANGUAGE I\_VOTES I\_GROSS

---------- -------------------------------------- ---------------------------------------- ---------- ---------- ----------

1014 2018:Everyone is a Hero 5/may/2023 Malayalam 97 750000000

1021 Oppenheimer 21/jul/2023 English 96 500000000

1032 Maamannan 29/jun/2023 Tamil 95 505000000

* **Insert data into the IndustryHit table by copying the appropriate columns in the Movies table for those Movies that have Votes greater than or equal to 95.**

**Query**

SQL> insert into industryhit (i\_id,i\_title,i\_releasedate,i\_language,i\_votes,i\_gross) select m\_id,title,releasedate,language,votes,gross from movies where votes >= 90;

5 rows created.

SQL> select \* from industryhit;

I\_ID I\_TITLE I\_RELEASEDATE I\_LANGUAGE I\_VOTES I\_GROSS

---------- -------------------------------------- -------------------------------------------------- ---------- ----------

1014 2018:Everyone is a Hero 5/may/2023 Malayalam 97 750000000

1021 Oppenheimer 21/jul/2023 English 96 500000000

1032 Maamannan 29/jun/2023 Tamil 95 505000000

1001 Hridayam 16-JUN-20 Malayalam 93 1600000000

1002 Meesa Madhavan 20-AUG-02 Malayalam 94 190000000

1007 3 idiots 25-DEC-09 Hindi 94 4600000000

1008 Chichchore 06-SEP-19 Hindi 91 1820000000

1010 Interstellar 07-NOV-14 English 92 7150000000

8 rows selected.

****

* **Movie Oppenheimer got a Metascore of 80. Make the appropriate data change.**

**Query**

-------------------[FIRST ADDING OPPENHEIMER TO TABLE : MOVIES]--------------------

SQL> insert into movies values(1021,'Oppenheimer','21/jul/2023','https://www.movies.com/oppenheimer.jpg','U/A',3,8.6,'Development of the atomic bomb.',75,96,'English',5500000000,1,'28/aug/2023');

1 row created.

//UPDATING METASCORE TO 80:

SQL> update movies set metascore=80 where m\_id=1021;

1 row updated.

* **Delete all movies whose Metascore is less than 50.**

**Query**

SQL> delete from movies where metascore < 50;

0 rows deleted.

* **Movie ‘Voice Of Sathyanathan’ was released.**

**For ‘Voice Of Sathyanathan’ enter the following data:**

**Id: 1015**

**Title: Voice Of Sathyanathan**

**Releasedate: 28 July 2023**

**Image\_url: https://m.media-amazon.com/imak2M\_.jpg**

**Certificate: U**

**Runtime: 2.10**

**ImdbRating: 7.4**

**Description: A man's life becomes increasingly complicated after his neighbor is injured in a dispute over a fence.**

**Metascore: 60**

**Votes: 90**

**Gross: 109500000**

**Query**

SQL> insert into movies values('1015','Voice Of Sathyanathan','18/jul/2023','https://m.media-amazon.com/imak2M\_.jpg','U',2.10,7.4,'A man life becomes increasing complicated after his neighbor is injured in a dispute over a fense',60,90,'Malayalam',109500000,0,'28/aug/2023');

1 row created.

* **Delete all rows from IndustryHit and drop the IndustryHit table.**

**Query**

SQL> delete from industryhit;

SQL> drop table industryhit;

Table dropped.

|  |  |
| --- | --- |
| **Description 3.2** | **Retrieval of data (Simple select query and select with ‘where’ options (include all relational and logical operators)** |
| **Date** | 14/08/2023 |

* **List details of all movies.**

**Query**

SQL> select \* from movies;

M\_ID TITLE RELEASEDA IMAGE\_URL CERTIFICATE RUNTIME IMDBRATING DESCRIPTION

METASCORE VOTES LANGUAGE GROSS HIT ENTRY\_DAT

---------- ---------------------------------------- --------- ---------------------------------------------------------------------------------------------------- -------------------- ---------- ---------- ---------------------------------------------------------------------------------------------------- ---------- ---------- -------------------- ---------- ---------- ---------

1001 Hridayam 16-JUN-20 https://www.movies.com/Hridayam.jpg U/A 2.34 8.4 The emotional journery of Arun

90 93 Malayalam 1600000000 1 28-AUG-23

1002 Meesa Madhavan 20-AUG-02 https://www.movies.com/Meesamadhavan.jpg U 2.45 8 Story of madhavan who is forced into a thief

92 94 Malayalam 190000000 1 28-AUG-23

1003 Wonder women 18-NOV-22 https://www.movies.com/wonderwomen.jpg U/A 1.2 5.2 story of six pregnant women

60 66 Malayalam 50000000 0 28-AUG-23

1004 Enthiran 01-OCT-10 https://www.movies.com/enthiran.jpg U/A 2.5 7.1 Story of humanoid robot

70 78 Tamil 3750000000 1 28-AUG-23

1005 Master 13-JAN-21 https://www.movies.com/master.jpg U/A 2.59 7.3 A professor clashes with a gangster

80 87 Tamil 2200000000 1 28-AUG-23

1006 Ponniyin Selvan:1 30-SEP-22 https://www.movies.com/ponniyinselvan1.jpg U/A 2.5 7.6 Chola Raja story

80 86 Tamil 3500000000 1 28-AUG-23

1007 3 idiots 25-DEC-09 https://www.movies.com/3idiots.jpg U/A 2.51 8.4 Story of 3 friends

90 94 Hindi 4600000000 1 28-AUG-23

1008 Chichchore 06-SEP-19 https://www.movies.com/chichchore.jpg U/A 2.23 8.3 life of college friends

90 91 Hindi 1820000000 1 28-AUG-23

1009 Avatar 18-DEC-09 https://www.movies.com/avatar.jpg U/A 2.42 7.9 Sci-fi epic

80 86 English 2930000000 1 28-AUG-23****

1010 Interstellar 07-NOV-14 https://www.movies.com/interstellar.jpg U/A 2.49 8.7 Ex-NASA pilot tasked to find new planet for humans

90 92 English 7150000000 1 28-AUG-23

1021 Oppenheimer 21-JUL-23 https://www.movies.com/oppenheimer.jpg U/A 3 8.6 Development of the atomic bomb.

80 96 English 5500000000 1 28-AUG-23

M\_ID TITLE RELEASEDA IMAGE\_URL CERTIFICATE RUNTIME IMDBRATING DESCRIPTION

METASCORE VOTES LANGUAGE GROSS HIT ENTRY\_DAT

---------- ---------------------------------------- --------- ---------------------------------------------------------------------------------------------------- -------------------- ---------- ---------- ---------------------------------------------------------------------------------------------------- ---------- ---------- -------------------- ---------- ---------- ---------

1015 Voice Of Sathyanathan 18-JUL-23 https://m.media-amazon.com/imak2M\_.jpg U 2.1 7.4 A man life becomes increasing complicated after his neighbor is injured in a dispute over a fense 60 90 Malayalam 109500000 0 28-AUG-23

12 rows selected.

* **List Title, Votes, Releasedate, Gross where Gross collection greater than 5000,000,00. Sequence the results in descending order by Gross.**

**Query**

SQL> select title,votes,releasedate,gross from movies where gross > 500000000 order by gross desc;

TITLE VOTES RELEASEDA GROSS

---------------------------------------- ---------- -------- ----------

Interstellar 92 07-NOV-14 7150000000

Oppenheimer 96 21-JUL-23 5500000000

3 idiots 94 25-DEC-09 4600000000****

Enthiran 78 01-OCT-10 3750000000

Ponniyin Selvan:1 86 30-SEP-22 3500000000

Avatar 86 18-DEC-09 2930000000

Master 87 13-JAN-21 2200000000

Chichchore 91 06-SEP-19 1820000000

Hridayam 93 16-JUN-20 1600000000

9 rows selected.

* **Retrieve the titles and years of Tamil movies released in 2022.**

**Query**

-------------------TWO POSSIBLE QUERIES---------------

SQL> select title,extract(year from releasedate) as YEAR from movies where language='Tamil' and (releasedate between '1/jan/2022' and '31/dec/2022');

TITLE YEAR

----------------------------------- --------

-----

Ponniyin Selvan:1 2022

>>OR

SQL> select title,extract(year from releasedate) as YEAR from movies where language='Tamil' and (extract(year from releasedate)='2022');

TITLE YEAR

---------------------------------------- ----------

Ponniyin Selvan:1 2022

* **Get the titles, years, and meta scores of movies sorted in descending order of meta scores.**

**Query**

SQL> select title,extract(year from releasedate) as YEAR,metascore from movies order by metascore desc;

TITLE YEAR METASCORE

---------------------------------------- ---------- ----------

Meesa Madhavan 2002 92

3 idiots 2009 90

Interstellar 2014 90

Chichchore 2019 90

Hridayam 2020 90

Master 2021 80

Ponniyin Selvan:1 2022 80

Oppenheimer 2023 80

Avatar 2009 80

Enthiran 2010 70

Wonder women 2022 60

TITLE YEAR METASCORE

---------------------------------------- ---------- ---------

Voice Of Sathyanathan 2023 60****

12 rows selected.

* **List titles, years, languages, dates and votes of all Malayalam and English movies released before 2022 and ImdbRating less than 7. The list should be ordered by Title.**

**Query**

------------[FIRST UPDATING ONE OF MOVIE VALUE TO BE LESS THAN 2022]------------

SQL> update movies set releasedate='18/nov/2021' where m\_id=1003;

1 row updated.

SQL> select title,extract(year from releasedate) as YEAR,language,releasedate,votes from movies where language in ('Malayalam','English') and extract(year from releasedate) < '2022' and imdbrating < 7 order by title;

TITLE YEAR LANGUAGE RELEASEDA VOTES

---------------------------------------- ---------- -------------------- --------- ----------

Wonder women 2021 Malayalam 18-NOV-21 66

* **List all the movies whose title starts with ‘Open’. Order the result by descending order of their id.**

**Query**

SQL> select m\_id,title from movies where title like('Open%') order by m\_id desc;

no rows selected

SQL> select m\_id,title from movies where title like('Oppen%') order by m\_id desc;

M\_ID TITLE

---------- ----------------------------------------****

1021 Oppenheimer

* **List Hit movies released in 2022 and 2023. Order the result by ascending order of their Titles.**

**Query**

----------------[TWO POSSIBLE QUERIES]---------------

SQL> select title as MOVIE from movies where hit=1 and extract(year from releasedate) in('2022','2023') order by title asc;

MOVIE

----------------------------------------

Oppenheimer

Ponniyin Selvan:1

>>OR

SQL> select title as MOVIE from movies where hit=1 and extract(year from releasedate) between '2022' and '2023' order by title asc;

MOVIE

----------------------------------------

Oppenheimer

Ponniyin Selvan:1

* **Retrieve movies with a runtime between 1.5 and 2.5 hours.**

**Query**

SQL> select title as movie\_name,runtime from movies where runtime between 1.5 and 2.5;

MOVIE\_NAME RUNTIME

---------------------------------------- ----------

Hridayam 2.34

Meesa Madhavan 2.45

Enthiran 2.5

Ponniyin Selvan:1 2.5

Chichchore 2.23

Avatar 2.42

Interstellar 2.49

Voice Of Sathyanathan 2.1

8 rows selected.

* **Retrieve movies with Metascore ratings below 50 and IMDb ratings above 6.0.**

**Query**

------------[FIRST UPDATING ONE OF METASCORE TO < 50]---------

SQL> update movies set metascore = 45 where m\_id=1004;

1 row updated.

SQL> select title as movie\_name,metascore from movies where metascore < 50 and imdbrating > 6.0;

MOVIE\_NAME METASCORE

---------------------------------------- ----------

Enthiran 45

* **Retrieve movies with no description provided.**

**Query**

-----------------[FIRST UPDATING ONE OF DESCRIPTION TO NULL]-------------

SQL> update movies set description = null where m\_id = 1006;

1 row updated.

SQL> select title as movie\_name from movies where description is null;

MOVIE\_NAME

----------------------------------------

Ponniyin Selvan:1

****

|  |  |
| --- | --- |
| **Description 3.3** | **Functions: Numeric Data, Character Conversion and Group functions.** |
| **Date** | 14/08/2023 |

* **Illustrate the different numeric functions using dual table (power,round, ceil, floor, abs, exp, greatest, least, mod, trunc, round,sign, sqrt etc.)**

**Query**

**//POWER:-**

SQL> select power(2,3) from dual;

POWER(2,3)

----------

8

**//ROUND:-**

SQL> select round(12.345,2) from dual;

ROUND(12.345,2)

---------------

12.35

**//CEIL:-**

SQL> select ceil(12.345) from dual;

****

CEIL(12.345)

------------

13

**//FLOOR:-**

SQL> select floor(12.345) from dual;

FLOOR(12.345)

-------------

12

**//ABS:-**

SQL> select abs(-12.345) from dual;

ABS(-12.345)

------------

12.345

**//EXP:-**

SQL> select exp(2) from dual;

EXP(2)

----------

7.3890561

**//GREATEST:-**

SQL> select greatest(1,2,3) from dual;

GREATEST(1,2,3)

---------------

3

**//LEAST:-**

SQL> select least(1,2,3) from dual;

LEAST(1,2,3)

------------

1

**//MOD:-**

SQL> select mod(10,3) from dual;

MOD(10,3)

----------

1

**//TRUNC:-**

SQL> select trunc(12.345,1) from dual;

TRUNC(12.345,1)

---------------

12.3

**//SIGN:-**

SQL> select sign(-12.345) from dual;

SIGN(-12.345)

-------------

-1

**//SQRT:-**

SQL> select sqrt(16) from dual;

SQRT(16)

----------

4

* **Illustrate the character functions (upper, lower, initcap, length,concat, ascii, substr, ltrim, rtrim, trim, translate, instr,chr,Lpad,Rpadetc) using the table Movies.**

**//UPPER:-**

**Query**

SQL> select title,upper(title) from movies;

TITLE UPPER(TITLE)

---------------------------------------- ----------------------------------------

Hridayam HRIDAYAM

Meesa Madhavan MEESA MADHAVAN

Wonder women WONDER WOMEN

Enthiran ENTHIRAN

Master MASTER****

Ponniyin Selvan:1 PONNIYIN SELVAN:1

3 idiots 3 IDIOTS

Chichchore CHICHCHORE

Avatar AVATAR

Interstellar INTERSTELLAR

Oppenheimer OPPENHEIMER

TITLE UPPER(TITLE)

---------------------------------------- ----------------------------------------

Voice Of Sathyanathan VOICE OF SATHYANATHAN

12 rows selected.

**//LOWER:-**

**Query**

SQL> select title,lower(title) from movies;

TITLE LOWER(TITLE)

---------------------------------------- ----------------------------------------

Hridayam hridayam

Meesa Madhavan meesa madhavan

Wonder women wonder women

Enthiran enthiran

Master master

Ponniyin Selvan:1 ponniyin selvan:1

3 idiots 3 idiots

Chichchore chichchore

Avatar avatar

Interstellar interstellar

Oppenheimer oppenheimer

TITLE LOWER(TITLE)

---------------------------------------- ----------------------------------------

Voice Of Sathyanathan voice of sathyanathan

12 rows selected.

**//INITCAP:-**

**Query**

->The title of the movie with the first letter of each word capitalized

SQL> select title,initcap(title) from movies;

TITLE INITCAP(TITLE)

---------------------------------------- ----------------------------------------

Hridayam Hridayam

Meesa Madhavan Meesa Madhavan

Wonder women Wonder Women

Enthiran Enthiran

Master Master

Ponniyin Selvan:1 Ponniyin Selvan:1

3 idiots 3 Idiots

Chichchore Chichchore

Avatar Avatar

Interstellar Interstellar

Oppenheimer Oppenheimer

TITLE INITCAP(TITLE)

---------------------------------------- ----------------------------------------

Voice Of Sathyanathan Voice Of Sathyanathan

12 rows selected.

**//LENGTH:-**

**Query**

-> The length of the title of the movie:

SQL> select title,length(title) from movies;

TITLE LENGTH(TITLE)

---------------------------------------- -------------

Hridayam 8

Meesa Madhavan 14

Wonder women 12

Enthiran 8

Master 6****

Ponniyin Selvan:1 17

3 idiots 8

Chichchore 10

Avatar 6

Interstellar 12

Oppenheimer 11

TITLE LENGTH(TITLE)

---------------------------------------- -------------

Voice Of Sathyanathan 21

12 rows selected.

**//CONCAT:-**

**Query**

->The title of the movie concatenated with the language.

SQL> select title,concat(title,language) from movies;

TITLE CONCAT(TITLE,LANGUAGE)

---------------------------------------- ------------------------------------------------------------

Hridayam HridayamMalayalam

Meesa Madhavan Meesa MadhavanMalayalam

Wonder women Wonder womenMalayalam

Enthiran EnthiranTamil

Master MasterTamil

Ponniyin Selvan:1 Ponniyin Selvan:1Tamil

3 idiots 3 idiotsHindi

Chichchore ChichchoreHindi

Avatar AvatarEnglish

Interstellar InterstellarEnglish

Oppenheimer OppenheimerEnglish

TITLE CONCAT(TITLE,LANGUAGE)

---------------------------------------- ------------------------------------------------------------

Voice Of Sathyanathan Voice Of SathyanathanMalayalam

12 rows selected.****

**//ASCII:-**

**Query**

->The ASCII code for the first letter is displayed:

SQL> select title,ASCII(title) from movies;

TITLE ASCII(TITLE)

---------------------------------------- ------------

Hridayam 72

Meesa Madhavan 77

Wonder women 87

Enthiran 69

Master 77

Ponniyin Selvan:1 80

3 idiots 51

Chichchore 67

Avatar 65

Interstellar 73

Oppenheimer 79

TITLE ASCII(TITLE)

---------------------------------------- ------------

Voice Of Sathyanathan 86

12 rows selected.

**//SUBSTR:-**

**Query**

->The first 3 characters of the title of the movie are:

SQL> select title,substr(title,1,3) from movies;

TITLE SUB

----------------------------- -------------

Hridayam Hri

Meesa Madhavan Mee

Wonder women Won

Enthiran Ent

Master Mas

Ponniyin Selvan:1 Pon

3 idiots 3 i

Chichchore Chi

Avatar Ava

Interstellar Int

Oppenheimer Opp****

TITLE SUB

------------------------------------ ---- ---

Voice Of Sathyanathan Voi

12 rows selected.

**//LTRIM:-**

**Query**

->The title of the movie with leading spaces trimmed:

SQL> select title,ltrim(title) from movies;

TITLE LTRIM(TITLE)

---------------------------------------- ----------------------------------------

Hridayam Hridayam

Meesa Madhavan Meesa Madhavan

Wonder women Wonder women

Enthiran Enthiran

Master Master

Ponniyin Selvan:1 Ponniyin Selvan:1

3 idiots 3 idiots

Chichchore Chichchore

Avatar Avatar

Interstellar Interstellar

Oppenheimer Oppenheimer

TITLE LTRIM(TITLE)

---------------------------------------- ----------------------------------------

Voice Of Sathyanathan Voice Of Sathyanathan

12 rows selected.

SQL> select ltrim(' hello') from dual;

LTRIM

-----

hello

**//RTRIM:**

**Query**

->The title of the movie with trailing spaces trimmed:

SQL> select title,rtrim(title) from movies;

TITLE RTRIM(TITLE)

---------------------------------------- ----------------------------------------

Hridayam Hridayam

Meesa Madhavan Meesa Madhavan****

Wonder women Wonder women

Enthiran Enthiran

Master Master

Ponniyin Selvan:1 Ponniyin Selvan:1

3 idiots 3 idiots

Chichchore Chichchore

Avatar Avatar

Interstellar Interstellar

Oppenheimer Oppenheimer

TITLE RTRIM(TITLE)

---------------------------------------- ----------------------------------------

Voice Of Sathyanathan Voice Of Sathyanathan

12 rows selected.

SQL> select rtrim(' hello ') from dual;

RTRIM('H

--------

hello

**//TRIM :-**

**Query**

->The title of the movie with leading and trailing spaces trimmed:

SQL> select title,trim(title) from movies;

TITLE TRIM(TITLE)

---------------------------------------- ----------------------------------------

Hridayam Hridayam

Meesa Madhavan Meesa Madhavan

Wonder women Wonder women

Enthiran Enthiran

Master Master

Ponniyin Selvan:1 Ponniyin Selvan:1

3 idiots 3 idiots

Chichchore Chichchore

Avatar Avatar

Interstellar Interstellar

Oppenheimer Oppenheimer

****

TITLE TRIM(TITLE)

---------------------------------------- ----------------------------------------

Voice Of Sathyanathan Voice Of Sathyanathan

12 rows selected.

SQL> select trim(' hello ') from dual;

TRIM(

-----

hello

**//TRANSLATE :-**

**Query**

The title of the movie with all the letters "a" will be replaced by "z":

SQL> select title,translate(title,'a','z') from movies;

TITLE TRANSLATE(TITLE,'A','Z')

---------------------------------------- ----------------------------------------

Hridayam Hridzyzm

Meesa Madhavan Meesz Mzdhzvzn

Wonder women Wonder women

Enthiran Enthirzn

Master Mzster

Ponniyin Selvan:1 Ponniyin Selvzn:1

3 idiots 3 idiots

Chichchore Chichchore

Avatar Avztzr

Interstellar Interstellzr

Oppenheimer Oppenheimer

TITLE TRANSLATE(TITLE,'A','Z')

---------------------------------------- ----------------------------------------

Voice Of Sathyanathan Voice Of Szthyznzthzn

12 rows selected.

**//INSTR:-**

**Query**

->The position of the substring "a" in the title of the movie is :

SQL> select title,instr(title,'a') from movies;

TITLE INSTR(TITLE,'A')

---------------------------------------- ----------------

Hridayam 5

Meesa Madhavan 5

Wonder women 0

Enthiran 7

Master 2

Ponniyin Selvan:1 14

3 idiots 0

Chichchore 0

Avatar 3

Interstellar 11

Oppenheimer 0

TITLE INSTR(TITLE,'A')

---------------------------------------- ----------------

Voice Of Sathyanathan 11

12 rows selected.

**//CHR:-**

**Query**

SQL> select votes,chr(votes) from movies;

VOTES C

---------- -

93 ]****

94 ^

66 B

78 N

87 W

86 V

94 ^

91 [

86 V

92 \

96 `

VOTES C

---------- -

90 Z

12 rows selected.

**//LPAD:-**

**Query**

-> The title of the movie padded with specific number of \* to the left:

SQL> select title,lpad(title,20,'\*') from movies;

TITLE LPAD(TITLE,20,'\*')

---------------------------------------- --------------------

Hridayam \*\*\*\*\*\*\*\*\*\*\*\*Hridayam

Meesa Madhavan \*\*\*\*\*\*Meesa Madhavan

Wonder women \*\*\*\*\*\*\*\*Wonder women

Enthiran \*\*\*\*\*\*\*\*\*\*\*\*Enthiran

Master \*\*\*\*\*\*\*\*\*\*\*\*\*\*Master

****Ponniyin Selvan:1 \*\*\*Ponniyin Selvan:1

3 idiots \*\*\*\*\*\*\*\*\*\*\*\*3 idiots

Chichchore \*\*\*\*\*\*\*\*\*\*Chichchore

Avatar \*\*\*\*\*\*\*\*\*\*\*\*\*\*Avatar

Interstellar \*\*\*\*\*\*\*\*Interstellar

Oppenheimer \*\*\*\*\*\*\*\*\*Oppenheimer

TITLE LPAD(TITLE,20,'\*')

---------------------------------------- --------------------

Voice Of Sathyanathan Voice Of Sathyanatha

12 rows selected.

**//RPAD:-**

**Query**

->The title of the movie padded with specific number of \* to the right:

SQL> select title,rpad(title,20,'\*') from movies;

TITLE RPAD(TITLE,20,'\*')

---------------------------------------- --------------------

Hridayam Hridayam\*\*\*\*\*\*\*\*\*\*\*\*

Meesa Madhavan Meesa Madhavan\*\*\*\*\*\*

Wonder women Wonder women\*\*\*\*\*\*\*\*

Enthiran Enthiran\*\*\*\*\*\*\*\*\*\*\*\*

Master Master\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Ponniyin Selvan:1 Ponniyin Selvan:1\*\*\*

3 idiots 3 idiots\*\*\*\*\*\*\*\*\*\*\*\*

Chichchore Chichchore\*\*\*\*\*\*\*\*\*\*

Avatar Avatar\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Interstellar Interstellar\*\*\*\*\*\*\*\*

Oppenheimer Oppenheimer\*\*\*\*\*\*\*\*\*

TITLE RPAD(TITLE,20,'\*')

---------------------------------------- -------------------

Voice Of Sathyanathan Voice Of Sathyanatha

12 rows selected.

* **Illustration of conversion functions- to\_number,to\_char(numberconversion), to\_char(dateconversion)**

**//TO\_NUMBER :-**

**Query**

->This code will first convert the string '12345' to a number. The result will be a number with the data type NUMBER.

SQL> select TO\_NUMBER('12345') from dual;

TO\_NUMBER('12345')

------------------

12345

**//TO\_CHAR (NUMBER CONVERSION):-**

**Query**

SQL> SELECT TO\_CHAR(75917.63,'$99,999.99') from dual;

TO\_CHAR(759

-----------

$75,917.63

SQL> select gross,TO\_CHAR(gross,'$999,99,99,999.99') from movies;

GROSS TO\_CHAR(GROSS,'$99

---------- ------------------

1600000000 $160,00,00,000.00

190000000 $19,00,00,000.00

50000000 $5,00,00,000.00

3750000000 $375,00,00,000.00****

2200000000 $220,00,00,000.00

3500000000 $350,00,00,000.00

4600000000 $460,00,00,000.00

1820000000 $182,00,00,000.00

2930000000 $293,00,00,000.00

7150000000 $715,00,00,000.00

5500000000 $550,00,00,000.00

GROSS TO\_CHAR(GROSS,'$99

---------- ------------------

109500000 $10,95,00,000.00

12 rows selected.

**//TO\_CHAR (DATE CONVERSION):-**

**Query**

SQL> select sysdate,TO\_CHAR(sysdate,'day') from dual;

SYSDATE TO\_CHAR(S

--------- ---------

29-AUG-23 tuesday

SQL> select releasedate,TO\_CHAR(releasedate,'ddth-mon-yy') as DAY from movies;

RELEASEDA DAY

--------- -----------

16-JUN-20 16th-jun-20

20-AUG-02 20th-aug-02

18-NOV-21 18th-nov-21

01-OCT-10 01st-oct-10

13-JAN-21 13th-jan-21

30-SEP-22 30th-sep-22

25-DEC-09 25th-dec-09

06-SEP-19 06th-sep-19

18-DEC-09 18th-dec-09

07-NOV-14 07th-nov-14

21-JUL-23 21st-jul-23

RELEASEDA DAY

--------- -----------

18-JUL-23 18th-jul-23

12 rows selected.

****

* **Count the total no. of Movies**

**Query**

SQL> select COUNT(\*) as Total\_Movies from movies;

TOTAL\_MOVIES

------------

12

SQL> select COUNT(m\_id) as Total\_Movies from movies;

TOTAL\_MOVIES

------------

12

* **Calculate the average votes of movies.**

**Query**

SQL> select AVG(votes) from movies;

AVG(VOTES)

----------

87.75

* **Determine the maximum and minimum collection of movies. Rename the output as Max\_Coll and Min\_Coll respectively.**

**Query**

SQL> select MAX(gross) as Max\_Coll,MIN(gross) as Min\_Col from movies;

MAX\_COLL MIN\_COL

---------- ----------

7150000000 50000000

* **Count the number of movies crossed the collection 50,00,00,000.**

**Query**

SQL> select COUNT(\*) as movies\_crossed from movies where gross > 500000000;

MOVIES\_CROSSED

--------------

9

* **Count the hit movies of 2021.**

**Query**

SQL> select COUNT(\*) as hit\_movies from movies where hit=1 and extract(year from releasedate) = 2021;

HIT\_MOVIES

----------

1****

|  |  |
| --- | --- |
| **Description 3.4** | **Data manipulations using date functions** |
| **Date** | 14/08/2023 |

* **Provide a list of all movies which were released on June 16, 2020.Display the year and month of the released date and the Id. Sort the result by Id. Name the derived columns YEAR and MONTH.**

**Query**

SQL> select m\_id,title,TO\_CHAR(releasedate,'yyyy') as YEAR,TO\_CHAR(releasedate,'month') as MONTH from movies where releasedate='16/jun/2020' order by m\_id;

M\_ID TITLE YEAR MONTH

---------- ---------------------------------------- ---- ---------

1001 Hridayam 2020 june

* **List the number of months between release date and entry date of each movie.**

**Query**

SQL> select m\_id,title,releasedate,entry\_date,MONTHS\_BETWEEN(entry\_date,releasedate) as NO\_OF\_MONTHS\_BETWEEN from movies;

M\_ID TITLE RELEASEDA ENTRY\_DAT NO\_OF\_MONTHS\_BETWEEN

---------- ---------------------------------------- --------- --------- --------------------

1001 Hridayam 16-JUN-20 28-AUG-23 38.3870968

1002 Meesa Madhavan 20-AUG-02 28-AUG-23 252.258065

1003 Wonder women 18-NOV-21 28-AUG-23 21.3225806

1004 Enthiran 01-OCT-10 28-AUG-23 154.870968

1005 Master 13-JAN-21 28-AUG-23 31.483871

1006 Ponniyin Selvan:1 30-SEP-22 28-AUG-23 10.9354839

1007 3 idiots 25-DEC-09 28-AUG-23 164.096774

1008 Chichchore 06-SEP-19 28-AUG-23 47.7096774

1009 Avatar 18-DEC-09 28-AUG-23 164.322581

1010 Interstellar 07-NOV-14 28-AUG-23 105.677419

1021 Oppenheimer 21-JUL-23 28-AUG-23 1.22580645

M\_ID TITLE RELEASEDA ENTRY\_DAT NO\_OF\_MONTHS\_BETWEEN

---------- ---------------------------------------- --------- --------- --------------------

1015 Voice Of Sathyanathan 18-JUL-23 28-AUG-23 1.32258065

****

12 rows selected.

* **List the Entry\_date in the format ‘DD-Month-YY’.**

**Query**

SQL> select m\_id,TO\_CHAR(entry\_date,'DD-month-YY') from movies;

M\_ID TO\_CHAR(ENTRY\_D

---------- ---------------

1001 28-august -23

1002 28-august -23

1003 28-august -23

1004 28-august -23

1005 28-august -23

1006 28-august -23

1007 28-august -23

1008 28-august -23

1009 28-august -23

1010 28-august -23

1021 28-august -23

M\_ID TO\_CHAR(ENTRY\_D

---------- ---------------

1015 28-august -23

12 rows selected.

* **List the date, 8 days after today’s date.**

**Query**

SQL> select sysdate+8 from dual;

****

SYSDATE+8

---------

06-SEP-23

* **List all the movies which were released in the month of February.**

**Query**

------------------[FIRST UPDATING ONE OF MOVIES's RELEASED MONTH TO FEB]---------------

SQL> update movies set releasedate='20/feb/2002' where m\_id=1002;

1 row updated.

SQL> select m\_id,title from movies where TO\_CHAR(releasedate,'MON') = 'FEB';

M\_ID TITLE

---------- ----------------------------------------

1002 Meesa Madhavan

* **Illustrate the different date functions using dual table (to\_date,Add\_months, last\_day, months\_between, next\_day, round etc.)**

**//TO\_DATE :-**

**Query**

SQL> select TO\_DATE('2023-08-29','YYYY-MM-DD') from dual;

TO\_DATE('

---------

29-AUG-23

**//ADD\_MONTHS:-**

**Query**

SQL> select sysdate,ADD\_MONTHS(sysdate,4) from dual;

SYSDATE ADD\_MONTH

--------- ---------

29-AUG-23 29-DEC-23

SQL>

**//LAST\_DAY:-**

**Query**

SQL> select sysdate,LAST\_DAY(sysdate) from dual;

SYSDATE LAST\_DAY(****

--------- ---------

29-AUG-23 31-AUG-23

**//MONTHS\_BETWEEN:-**

**Query**

SQL> select MONTHS\_BETWEEN('25-AUG-23','25-DEC-22') from dual;

MONTHS\_BETWEEN('25-AUG-23','25-DEC-22')

---------------------------------------

8

**//NEXT\_DAY:-**

**Query**

SQL> select sysdate,NEXT\_DAY(sysdate,'FRIDAY') from dual;

SYSDATE NEXT\_DAY(

--------- ---------

29-AUG-23 01-SEP-23

**//ROUND:-**

**Query**

SQL> select sysdate,ROUND(sysdate,'MM') as nearest\_month from dual;

SYSDATE NEAREST\_M

--------- ---------

29-AUG-23 01-SEP-23

* **Illustration of special date formats using to\_char function (use of th,sp,spth)**

**//TO\_CHAR(TH):-**

**Query**

SQL> select sysdate,TO\_CHAR(sysdate,'ddth-mon-yy') from dual;

SYSDATE TO\_CHAR(SYS

--------- -----------

29-AUG-23 29th-aug-23

SQL> select releasedate,TO\_CHAR(releasedate,'ddth-mon-yy') as DAY from movies;

RELEASEDA DAY

--------- -----------

16-JUN-20 16th-jun-20

20-FEB-02 20th-feb-02****

18-NOV-21 18th-nov-21

01-OCT-10 01st-oct-10

13-JAN-21 13th-jan-21

30-SEP-22 30th-sep-22

25-DEC-09 25th-dec-09

06-SEP-19 06th-sep-19

18-DEC-09 18th-dec-09

07-NOV-14 07th-nov-14

21-JUL-23 21st-jul-23

RELEASEDA DAY

--------- -----------

18-JUL-23 18th-jul-23

12 rows selected.

**//TO\_CHAR(SP):-**

**Query**

SQL> select sysdate,TO\_CHAR(sysdate,'ddsp-mon-yy') from dual;

SYSDATE TO\_CHAR(SYSDATE,'DD

--------- -------------------

29-AUG-23 twenty-nine-aug-23

SQL> select releasedate,TO\_CHAR(releasedate,'ddsp-mon-yy') as DAY from movies;

RELEASEDA DAY

--------- -------------------

16-JUN-20 sixteen-jun-20

20-FEB-02 twenty-feb-02

18-NOV-21 eighteen-nov-21

01-OCT-10 one-oct-10

13-JAN-21 thirteen-jan-21

30-SEP-22 thirty-sep-22

25-DEC-09 twenty-five-dec-09

06-SEP-19 six-sep-19

18-DEC-09 eighteen-dec-09

07-NOV-14 seven-nov-14

21-JUL-23 twenty-one-jul-23

****

RELEASEDA DAY

--------- -------------------

18-JUL-23 eighteen-jul-23

12 rows selected.

**//TO\_CHAR(SPTH):-**

**Query**

SQL> select sysdate,TO\_CHAR(sysdate,'ddspth-mon-yy') from dual;

SYSDATE TO\_CHAR(SYSDATE,'DDSP

--------- ---------------------

29-AUG-23 twenty-ninth-aug-23

SQL> select releasedate,TO\_CHAR(releasedate,'ddspth-mon-yy') as DAY from movies;

RELEASEDA DAY

--------- ---------------------

16-JUN-20 sixteenth-jun-20

20-FEB-02 twentieth-feb-02

18-NOV-21 eighteenth-nov-21

01-OCT-10 first-oct-10

13-JAN-21 thirteenth-jan-21

30-SEP-22 thirtieth-sep-22

25-DEC-09 twenty-fifth-dec-09

06-SEP-19 sixth-sep-19

18-DEC-09 eighteenth-dec-09

07-NOV-14 seventh-nov-14

****21-JUL-23 twenty-first-jul-23

RELEASEDA DAY

--------- ---------------------

18-JUL-23 eighteenth-jul-23

12 rows selected.

* **Calculate the total gross earnings for movies released after June 16, 2020.**

**Query**

SQL> select SUM(GROSS) from movies where releasedate>'16/jun/2020';

SUM(GROSS)

----------

1.1360E+10

|  |  |
| --- | --- |
| **Description 3.5** | **Set Operations** |
| **Date** | 14/08/2023 |

* **Create a new table IndustryHit (Id, title, genre, Certificate, Gross, Releasedate). Insert some movies from Movies table and some new movies in the new table IndustryHit.**

**Query**

****

SQL> create table industryhit(i\_id int,i\_title varchar2(40),genre varchar2(40),certificate varchar2(20),gross number(12,2),releasedate date,constraint prky\_iid primary key(i\_id));

Table created.

SQL> desc industryhit;

Name Null? Type

----------------------------------------------------- -------- ------------------------------------

I\_ID NOT NULL NUMBER(38)

I\_TITLE VARCHAR2(40)

GENRE VARCHAR2(40)

CERTIFICATE VARCHAR2(20)

GROSS NUMBER(12,2)

RELEASEDATE DATE

SQL> select constraint\_name,constraint\_type from user\_constraints where table\_name='INDUSTRYHIT';

CONSTRAINT\_NAME C

--------------------------- --- -

PRKY\_IID P

SQL>

**//INSERTING:**

SQL> insert into industryhit (select m\_id,title,description,certificate,gross,releasedate from movies where m\_id=1001 or m\_id=1004 or m\_id=1007 or m\_id=1009 or m\_id=1021);

5 rows created.

SQL> select \* from industryhit;

I\_ID I\_TITLE GENRE CERTIFICATE GROSS RELEASEDA

---------- ---------------------------------------- ---------------------------------------- -------------------- ---------- ---------

1001 Hridayam The emotional journery of Arun U/A 1600000000 16-JUN-20

1004 Enthiran Story of humanoid robot U/A 3750000000 01-OCT-10

1007 3 idiots Story of 3 friends U/A 4600000000 25-DEC-09

1009 Avatar Sci-fi epic U/A 2930000000 18-DEC-09

1021 Oppenheimer Development of the atomic bomb. U/A 5500000000 21-JUL-23

SQL> insert into industryhit values(1031,'Mission Impossible - Fallout','Action Thriller','U/A',7910000000,'27/jul/2018');

1 row created.

SQL> insert into industryhit values(1032,'Premam','Romance/Drama','U',760000000,'29/may/2015');

1 row created.

SQL> insert into industryhit values(1033,'Dangal','Action/Sport','U',5380000000,'23/Dec/2016');

1 row created.

SQL> select \* from industryhit;

I\_ID I\_TITLE GENRE CERTIFICATE GROSS RELEASEDA****

---------- ---------------------------------------- ---------------------------------------- -------------------- ---------- -----

1001 Hridayam The emotional journery of Arun U/A 1600000000 16-JUN-20

1004 Enthiran Story of humanoid robot U/A 3750000000 01-OCT-10

1007 3 idiots Story of 3 friends U/A 4600000000 25-DEC-09

1009 Avatar Sci-fi epic U/A 2930000000 18-DEC-09

1021 Oppenheimer Development of the atomic bomb. U/A 5500000000 21-JUL-23

1031 Mission Impossible - Fallout Action Thriller U/A 7910000000 27-JUL-18

1032 Premam Romance/Drama U 760000000 29-MAY-15

1033 Dangal Action/Sport U 5380000000 23-DEC-16

8 rows selected.

* **Retrieve the titles of all movies and industry hits which are in the action thriller genre.**

**Query**

SQL> select title from movies UNION select i\_title from industryhit where genre='Action Thriller';

TITLE

----------------------------------------

3 idiots

Avatar

Chichchore

Enthiran

Hridayam

Interstellar

Master

Meesa Madhavan

Mission Impossible - Fallout

Oppenheimer

Ponniyin Selvan:1

****TITLE

----------------------------------------

Voice Of Sathyanathan

Wonder women

13 rows selected.

* **Retrieve the titles of all movies including industry hits.**

**Query**

SQL> select title from movies UNION select i\_title from industryhit;

TITLE

----------------------------------------

3 idiots

Avatar

Chichchore

Dangal

Enthiran

Hridayam

Interstellar

Master

Meesa Madhavan

Mission Impossible - Fallout

Oppenheimer

TITLE

----------------------------------------

Ponniyin Selvan:1

Premam

Voice Of Sathyanathan

Wonder women

15 rows selected.

* **Retrieve the titles of all movies which are not industry hits.**

**Query**

SQL> select title from movies MINUS select i\_title from industryhit;

TITLE

----------------------------------------

Chichchore

Interstellar

Master

Meesa Madhavan

Ponniyin Selvan:1

Voice Of Sathyanathan

Wonder women

7 rows selected.

****

|  |  |
| --- | --- |
| **Description 3.6** | **Illustration of Group By having clause** |
| **Date** | 14/08/2023 |

* **For all genres, display genre type and the sum of all Gross for each genre. Name the derived column SUM\_COLL.**

**Query**

--------------[FIRST UPDATING VALUES OF GENRE TO GET SOME SIMILAR ONES]----------------

SQL> update industryhit set genre='Action Thriller' where i\_id=1004;

****1 row updated.

SQL> update industryhit set genre='Romance/Drama' where i\_id=1001;

1 row updated.

SQL> update industryhit set genre='Romance/Drama' where i\_id=1007;

1 row updated.

SQL> select genre,SUM(gross) as SUM\_COLL from industryhit group by genre;

GENRE SUM\_COLL

---------------------------------------- ----------

Development of the atomic bomb. 5500000000

Sci-fi epic 2930000000

Action/Sport 5380000000

Action Thriller 1.1660E+10

Romance/Drama 6960000000

* **For all genres, display the genre type and the number of titles. Name the derived column TITLE\_COUNT.**

**Query**

SQL> select genre,COUNT(i\_title) as TITLE\_COUNT from industryhit group by genre;

GENRE TITLE\_COUNT

---------------------------------------- -----------

Development of the atomic bomb. 1

Sci-fi epic 1

Action/Sport 1

Action Thriller 2

Romance/Drama 3

* **Display the genres which have more than 3 titles.**

**Query**

--------[FIRST INSERTING EXTRA ROW TO GET MORE THAN 3 COUNT FOR SAME GENRE]------

SQL> insert into industryhit values(1034,'Titanic','Romance/Drama','U/A',6740000000,'19/Dec/1997');

1 row created.

SQL> select genre,COUNT(i\_title) as TITLE\_COUNT from industryhit group by genre having COUNT(i\_title) > 3;

GENRE TITLE\_COUNT

---------------------------------------- -----------

Romance/Drama 4

* **Retrieve the total number of movies released in each year, only for years with at least 5 movies.**

**Query**

SQL> select TO\_CHAR(releasedate,'yyyy'),COUNT(i\_id) from industryhit group by TO\_CHAR(releasedate,'yyyy') having COUNT(i\_id) >= 5;

no rows selected

SQL> select TO\_CHAR(releasedate,'yyyy'),COUNT(i\_id) from industryhit group by TO\_CHAR(releasedate,'yyyy') having COUNT(i\_id) >= 2;

TO\_C COUNT(I\_ID)****

---- -----------

2009 2

* **List the certificates along with the number of movies for each certificate, but only show certificates with more than 3 movies.**

**Query**

SQL> select certificate,COUNT(i\_id) from industryhit group by certificate having COUNT(i\_id) > 3;

CERTIFICATE COUNT(I\_ID)

-------------------- -----------

U/A 7

* **Show the total gross earnings for each certificate, but only for certificates with total gross greater than $1 million.**

**Query**

SQL> select certificate,SUM(gross) from industryhit group by certificate having SUM(gross) > 1000000;

CERTIFICATE SUM(GROSS)

-------------------- ----------

U/A 3.3030E+10

U 6140000000

* **List the release years with the highest number of movies and the corresponding movie count, limited to the top 3 years.**

**Query**

**//INNER QUERY**

SQL> select to\_char(releasedate,'yyyy') year,count(i\_id) count from industryhit group by to\_char(releasedate,'yyyy') order by count(i\_id) desc;

YEAR COUNT

---- ----------

2009 2

2016 1

1997 1

2018 1

2015 1****

2020 1

2023 1

2010 1

8 rows selected.

**//FINAL QUERY**

SQL> select year,count from(select to\_char(releasedate,'yyyy') year,count(i\_id) count from industryhit group by to\_char(releasedate,'yyyy') order by count(i\_id) desc) where rownum<4;

YEAR COUNT

---- ----------

2009 2

2023 1

1997 1

|  |  |
| --- | --- |
| **Description 3.7** | **Sub queries** |
| **Date** | 14/08/2023 |

* **Retrieve the titles and runtime of movies with the highest Metascore.**

**Query**

SQL> select title,runtime from movies where metascore = (select MAX(metascore) from movies);

TITLE RUNTIME

---------------------------------------- ----------

Meesa Madhavan 2.45

* **List the titles of movies with a Gross amount greater than the average Gross amount of all movies.**

**Query**

**//INNER QUERY**

SQL> select AVG(gross) from movies;

AVG(GROSS)

----------

2783291667

**//FINAL QUERY**

SQL> select title,gross from movies where gross > (select AVG(gross) from movies);

TITLE GROSS

---------------------------------------- ----------

Enthiran 3750000000****

Ponniyin Selvan:1 3500000000

3 idiots 4600000000

Avatar 2930000000

Interstellar 7150000000

Oppenheimer 5500000000

6 rows selected.

* **Retrieve the titles and descriptions of movies with a Metascore lower than the average Metascore.**

**Query**

SQL> select title,metascore,description from movies where metascore < (select AVG(metascore) from movies);

TITLE METASCORE DESCRIPTION

---------------------------------------- ---------- -----------------

Wonder women 60 story of six pregnant women

Enthiran 45 Story of humanoid robot

Voice Of Sathyanathan 60 A man life becomes increasing complicated after his neighbor is injured in a dispute over a fense

* **List the movie titles and their IMDb ratings for movies released in the year with the highest average IMDb rating.**

**Query**

**//INNER QUERY:**

SQL> select max(avg(imdbrating)) from movies group by to\_char(releasedate,'yyyy');

MAX(AVG(IMDBRATING))

--------------------

8.7

**//INNER QUERY:**

SQL> select to\_char(releasedate,'yyyy') from movies group by to\_char(releasedate,'yyyy') having avg(imdbrating)=(select max(avg(imdbrating)) from movies group by to\_char(releasedate,'yyyy'));

TO\_C

----

2014

**//FINAL QUERY:**

SQL> select title,imdbrating from movies where to\_char(releasedate,'yyyy')=(select to\_char(releasedate,'yyyy') from movies group by to\_char(releasedate,'yyyy') having avg(imdbrating)=(select max(avg(imdbrating)) from movies group by to\_char(releasedate,'yyyy')));

TITLE IMDBRATING

---------------------------------------- ----------

Interstellar 8.7

****

* **Retrieve the movie titles and their IMDb ratings for movies that have a Metascore greater than twice their IMDb rating.**

**Query**

**//INNER QUERY:**

SQL> select 2\*imdbrating from movies;

2\*IMDBRATING

------------

16.8

16

10.4

14.2

14.6

15.2

16.8

16.6

15.8

17.4

17.2****

2\*IMDBRATING

------------

14.8

12 rows selected.

**//FINAL QUERY:**

SQL> select title,imdbrating from movies m where metascore > (select 2\*imdbrating from movies h where m.title=h.title);

TITLE IMDBRATING

---------------------------------------- ----------

Hridayam 8.4

Meesa Madhavan 8

Wonder women 5.2

Enthiran 7.1

Master 7.3

Ponniyin Selvan:1 7.6

3 idiots 8.4

Chichchore 8.3

Avatar 7.9

Interstellar 8.7

Oppenheimer 8.6

TITLE IMDBRATING

---------------------------------------- ----------

Voice Of Sathyanathan 7.4

12 rows selected.

* **Find the title and gross amount of the top 3 highest-grossing movies.**

**Query**

**//INNER QUERY:**

SQL> select max(gross) from movies;

MAX(GROSS)

----------

7150000000

**//INNER QUERY:**

SQL> select title,gross from movies m1 where gross=(select max(gross) from movies m2 where m1.title=m2.title)order by gross desc;

TITLE GROSS

---------------------------------------- ----------

Interstellar 7150000000

Oppenheimer 5500000000

3 idiots 4600000000

Enthiran 3750000000

Ponniyin Selvan:1 3500000000

Avatar 2930000000

Master 2200000000

Chichchore 1820000000

Hridayam 1600000000

Meesa Madhavan 190000000****

Voice Of Sathyanathan 109500000

TITLE GROSS

---------------------------------------- ----------

Wonder women 50000000

12 rows selected.

**//FINAL QUERY:**

SQL> select \* from(select title,gross from movies m1 where gross=(select max(gross) from movies m2 where m1.title=m2.title)order by gross desc) where rownum <= 3;

TITLE GROSS

---------------------------------------- ----------

Interstellar 7150000000

Oppenheimer 5500000000

3 idiots 4600000000

* **Calculate the total number of votes received by movies released in the year 2022.**

**Query**

-----------------[FIRST UPDATING ONE OF MOVIE YEAR TO 2022]----------

SQL> update movies set releasedate='6/sep/2022' where m\_id=1008;

1 row updated.

**//INNER QUERY:**

SQL> select votes,TO\_CHAR(releasedate,'yyyy') from movies where TO\_CHAR(releasedate,'yyyy')='2022';

VOTES TO\_C

---------- ----

86 2022

91 2022

****

**//FINAL QUERY:**

SQL> select sum(votes) from movies m1 where TO\_CHAR(releasedate,'yyyy')=(select TO\_CHAR(releasedate,'yyyy') from movies m2 where TO\_CHAR(releasedate,'yyyy')='2022' and m1.title=m2.title);

SUM(VOTES)

----------

177

* **List the titles and certificate ratings of movies that have an IMDb rating below the average IMDb rating.**

**Query**

**//INNER QUERY:**

SQL> select AVG(imdbrating) from movies;

AVG(IMDBRATING)

---------------

7.74166667

**//FINAL QUERY:**

SQL> select title,certificate,imdbrating from movies where imdbrating < (select AVG(imdbrating) from movies);

TITLE CERTIFICATE IMDBRATING

---------------------------------------- -------------------- ----------

Wonder women U/A 5.2

Enthiran U/A 7.1

Master U/A 7.3

Ponniyin Selvan:1 U/A 7.6

Voice Of Sathyanathan U 7.4

|  |  |
| --- | --- |
| **Description 3.8** | **Views** |
| **Date** | 14/08/2023 |

1. **Create a view called MovieDetails that combines information from the Movies, Directors, and Stars tables to display movie titles, directors' names, and the names of stars who acted in those movies.**

**Query**

SQL> create view moviedetails as select m.title as Title\_of\_movie,d.d\_name as Directors\_name,s.s\_name as Name\_of\_stars from movies m,directors d,stars s,moviesdirectors md,moviesstars ms where md.moviesid=m.m\_id and md.directorsid=d.d\_id and ms.moviesid=m.m\_id and ms.starsid=s.s\_id;

View created.

SQL> select \* from moviedetails;

TITLE\_OF\_MOVIE DIRECTORS\_NAME NAME\_OF\_STARS

---------------------------------------- ------------------------------ --------------------------------------

Hridayam VINEETH SREENIVASAN PRANAV MOHANLAL

Meesa Madhavan LAL JOSE DILEEP

Enthiran S SANKAR RAJINIKANTH

Master LOKESH KANAGARAJ VIJAY

Ponniyin Selvan:1 MANI RATNAM AISHWARYA RAI BACHCHAN

3 idiots RAJKUMAR HIRANI AAMIR KHAN

Chichchore NITESH TIWARI SUSHANT SINGH RAJPUT

Avatar JAMES CAMERON ZOE SALDANA

Interstellar CHRISTOPHER NOLAN MATTHEW MCCONAUGHEY

Wonder women ANJALI MENON PARVATHY THIRUVOTHU

10 rows selected.****

SQL>

1. **Create a view called HighlyRatedMovies that displays movies with IMDb ratings greater than 8.0, including their titles and ratings.**

**Query**

SQL> create view highlyratedmovies as select title,imdbrating from movies where imdbrating>8.0;

View created.

SQL> select \* from highlyratedmovies;

TITLE IMDBRATING

---------------------------------------- ----------

Hridayam 8.4

3 idiots 8.4

Chichchore 8.3

Interstellar 8.7

Oppenheimer 8.6

SQL>

1. **Create a view called DirectorMovies that provides a list of directors along with the number of movies they have directed.**

**Query**

SQL> create view directormovies as select d.d\_name as name\_of\_director,count(m.m\_id) number\_of\_movies from movies m,directors d,moviesdirectors md where md.moviesid=m.m\_id and md.directorsid=d.d\_id group by d.d\_name;

View created.

SQL> select \* from directormovies;

NAME\_OF\_DIRECTOR NUMBER\_OF\_MOVIES

------------------------------ ----------------

S SANKAR 1

ANJALI MENON 1

NITESH TIWARI 1****

LOKESH KANAGARAJ 1

LAL JOSE 1

MANI RATNAM 1

RAJKUMAR HIRANI 1

CHRISTOPHER NOLAN 1

VINEETH SREENIVASAN 1

JAMES CAMERON 1

10 rows selected.

SQL>

1. **Create a view called StarMovies that displays stars' names and the titles of movies they have acted in.**

**Query**

SQL> create view starmovies as select s.s\_name,m.title from stars s,movies m,moviesstars ms where ms.moviesid=m.m\_id and ms.starsid=s.s\_id ;

View created.

SQL> select \* from starmovies;

S\_NAME TITLE

---------------------------------------- ----------------------------------------

PRANAV MOHANLAL Hridayam

DILEEP Meesa Madhavan

RAJINIKANTH Enthiran

VIJAY Master

AISHWARYA RAI BACHCHAN Ponniyin Selvan:1

AAMIR KHAN 3 idiots

SUSHANT SINGH RAJPUT Chichchore

ZOE SALDANA Avatar

MATTHEW MCCONAUGHEY Interstellar

PARVATHY THIRUVOTHU Wonder women

10 rows selected.

****

SQL>

1. **Create a view called LongestMovies that lists the titles of movies with the longest runtimes (duration).**

**Query**

SQL> create view longestmovies as select title,runtime from movies where runtime=(select max(runtime) from movies);

View created.

SQL> select \* from longestmovies;

TITLE RUNTIME

---------------------------------------- ----------

Oppenheimer 3

SQL>

1. **Create a view called LanguageDistribution that shows the distribution of movies based on the languages they were released in, including the count of movies for each language.**

**Query**

SQL>

SQL> create view languagedistribution as select language,count(title) as number\_of\_movies from movies group by language;

View created.

SQL> select \* from languagedistribution;

LANGUAGE NUMBER\_OF\_MOVIES

-------------------- ----------------

Malayalam 4

Tamil 3

English 3

Hindi 2

SQL>

1. **Create a view called GrossEarnings that displays movies with their titles and gross earnings, sorted by earnings in descending order.**

**Query**

SQL> create view grossearnings as select title,gross from movies order by gross desc;

View created.

SQL> select \* from grossearnings;

TITLE GROSS

---------------------------- ------------ ----------

Interstellar 7150000000

Oppenheimer 5500000000

3 idiots 4600000000

Enthiran 3750000000****

Ponniyin Selvan:1 3500000000

Avatar 2930000000

Master 2200000000

Chichchore 1820000000

Hridayam 1600000000

Meesa Madhavan 190000000

Voice Of Sathyanathan 109500000

TITLE GROSS

------------------------------ ---------- ----------

Wonder women 50000000

12 rows selected.

SQL>

1. **Create a view called IndustryHitMovies that shows the titles and release dates of industry hit movies.**

**Query**

SQL> create view industryhitmovies as select title,releasedate from movies where hit=1;

View created.

SQL> select \* from industryhitmovies;

TITLE RELEASEDA

------------------------- --------------- ---------

Hridayam 16-JUN-20

Meesa Madhavan 20-FEB-02

Enthiran 01-OCT-10

Master 13-JAN-21

Ponniyin Selvan:1 30-SEP-22

3 idiots 25-DEC-09

Chichchore 06-SEP-22

Avatar 18-DEC-09

Interstellar 07-NOV-14

Oppenheimer 21-JUL-23

10 rows selected.

SQL>

1. **Create a view called MovieVotes that displays movies along with their titles and the number of votes they have received.**

**Query**

SQL> create view movievotes as select title,votes from movies;

View created.

SQL> select \* from movievotes;

TITLE VOTES

--------------------------- ------------- ----------

Hridayam 93

Meesa Madhavan 94

Wonder women 66

Enthiran 78

Master 87

Ponniyin Selvan:1 86

3 idiots 94

Chichchore 91

Avatar 86

Interstellar 92

Oppenheimer 96

TITLE VOTES****

------------------------------ ---------- ----------

Voice Of Sathyanathan 90

12 rows selected.

SQL>

1. **Create a view called CertifiedMovies that lists movies with their titles and certificates (e.g., U/A, U).**

**Query**

SQL> create view certifiedmovies as select title,certificate from movies;

View created.****

SQL> select \* from certifiedmovies;

TITLE CERTIFICATE

---------------------------------------- --------------------

Hridayam U/A

Meesa Madhavan U

Wonder women U/A

Enthiran U/A

Master U/A

Ponniyin Selvan:1 U/A

3 idiots U/A

Chichchore U/A

Avatar U/A

Interstellar U/A

Oppenheimer U/A

TITLE CERTIFICATE

---------------------------------------- --------------------

Voice Of Sathyanathan U

12 rows selected.

SQL>

|  |
| --- |
| **Activity #4** |
| **Practice PL/SQL** |

|  |  |
| --- | --- |
| **Description 4.1** | **Introduction to PL/SQL** |
| **Date** | 10/09/2023 |

1. **Write a PL/SQL code block to calculate the area of a circle for a value of radius varying from 3 to 7. Store the radius and corresponding values of calculated area in an empty table named Areas, consisting of two columns Radius and Area.**

**Query**

SQL> create table circle(radius number(5,2),area number(10,2));

Table created.

SQL> edit E:\plsql\_ans\p6\_area\_of\_circle.sql

**//PL/SQL**

[[

|  |
| --- |
| declare  i number;  a number;  begin  i:=3;  while i<=7  loop  a:=3.14\*i\*i;  insert into circle values(i,a);  i:=i+1;  end loop;  end; |

]]

SQL> @ E:\plsql\_ans\p6\_area\_of\_circle.sql

16 /

PL/SQL procedure successfully completed.

SQL> select \* from circle;

RADIUS AREA

---------- ----------

3 28.26

4 50.24

5 78.5

6 113.04

7 153.86

SQL>

1. **Write a PL/SQL block of code for inverting a number accepted from the console.**

**Query**

SQL> edit E:\plsql\_ans\p\_reverse.sql

**//PL/SQL**

[[

|  |
| --- |
| declare  num1 int;  num2 int;  begin  num1:=&num1;  num2:=0;  loop  num2:=num2\*10+mod(num1,10);  num1:=num1/10;  exit when num1=0;  end loop;  dbms\_output.put\_line('reverse is  '||num2);  end; |

]]

SQL> @ E:\plsql\_ans\p\_reverse.sql

16 /

|  |
| --- |
| Enter value for num1: 573  old   5: num1:=&num1;  new   5: num1:=573;  reverse is  375 |

1. **Write a PL/SQL code block that will accept an account number from the user and debit an amount of Rs.2000 from the account if the account has a minimum balance of 500 after the amount is debited. The process is fired on the Accounts table.**

**Query**

SQL> create table account(ac\_no number(15) primary key,ac\_name varchar(30),ac\_balance number(10,2));

Table created.

SQL> insert into account values(1001,'Nihal Muhammed',25000);

1 row created.

SQL> insert into account values(1004,'Majo augestine',39000);

1 row created.

SQL> insert into account values(1007,'Abhinav M S',59000);

****

1 row created.

SQL> insert into account values(1008,'sreekumar',45000);

1 row created.

SQL> insert into account values(1009,'Hari',1000);

1 row created.

SQL> select \* from account;

AC\_NO AC\_NAME AC\_BALANCE

---------- ------------------------- ----- ----------

1001 Nihal Muhammed 25000

1004 Majo augestine 39000

1007 Abhinav M S 59000

1008 sreekumar 45000

1009 Hari 1000

SQL> edit E:\plsql\_ans\p7\_account.sql

**//PL/SQL**

[[

|  |
| --- |
| declare  acno number(15);  bal number(10,2);  begin  acno:=&acno;  select ac\_balance into bal  from account where  ac\_no=acno;  if bal-2000<500 then  dbms\_output.put\_line('Transation not possible,Insufficient balance');  else  update account set ac\_balance=ac\_balance-2000 where ac\_no=acno;  dbms\_output.put\_line('Transaction Successfully completed');  end if;  end; |

]]

SQL> @ E:\plsql\_ans\p7\_account.sql

20 /

|  |
| --- |
| Enter value for acno: 1009  old 6: acno:=&acno;  new 6: acno:=1009;  Transation not possible,Insufficient balance |

PL/SQL procedure successfully completed.****

SQL> @ E:\plsql\_ans\p7\_account.sql

20 /

|  |
| --- |
| Enter value for acno: 1001  old 6: acno:=&acno;  new 6: acno:=1001;  Transaction Successfully completed |

PL/SQL procedure successfully completed.

SQL> select \* from account;

AC\_NO AC\_NAME AC\_BALANCE

---------- ------------------------- ----- ----------

1001 Nihal Muhammed 23000

1004 Majo augestine 39000

1007 Abhinav M S 59000

1008 sreekumar 45000

1009 Hari 1000

****

SQL>

1. **Write a PL/SQL block of code that updates the salaries of Maria Jacob and Albert by Rs. 2000/- and Rs.2500/- respectively. Then check to see that the total salary does not exceed 75000. If the total salary is greater than 75000, then undo the updates made to salaries of both. (Use savepoint, rollback and commit).**

**Query**

SQL> create table employ(empno int,name varchar2(20),salary number(10,2));

Table created.

SQL> insert into employ values(101,'Maria',20000);

1 row created.

SQL> insert into employ values(102,'Albert',15000);

1 row created.

SQL> insert into employ values(103,'Megha',20000);

1 row created.

SQL> edit E:\plsql\_ans\p\_maria.sql

**//PL/SQL**

[[

|  |
| --- |
| declare  s number(10,2);  Begin  savepoint s1;  update employ set salary=salary+2000 where empno=101 or empno=102;  select sum(salary) into s from employ;  if s >75000  Then  rollback to s1;  dbms\_output.put\_line('Rollbacked');  Else  commit;  end if;  end; |

]]

SQL> @ E:\plsql\_ans\p\_maria.sql

PL/SQL procedure successfully completed.

****

|  |  |
| --- | --- |
| **Description 4.2.1** | **Illustration of Implicit Cursors.** |
| **Date** | 10/09/2023 |

1. **Write a PL/SQL block to accept an employee number and update the salary of that employee to raise the salary by 0.15. Display appropriate message based on the existence of the record in the employee table.**

**Query**

****

**//BEFORE**

SQL> select empno,salary from employee where empno='E0100';

EMPNO SALARY

---------- ----------

E0100 46000

SQL> edit E:\plsql\_ans\cursor\_q1\_HRD.sql

[[

|  |
| --- |
| declare  begin  update employee set salary=salary+salary\*0.15 where empno=&empno;  if sql%found then  dbms\_output.put\_line(sql%rowcount ||' SALARY HAS BEEN UPDATED');  else  dbms\_output.put\_line('NO RECORDS UPDATED');  end if;  end; |

]]

SQL> @ E:\plsql\_ans\cursor\_q1\_HRD.sql

15 /

|  |
| --- |
| Enter value for empno: 'E0100'  old 5: update employee set salary=salary+salary\*0.15 where empno=&empno;  new 5: update employee set salary=salary+salary\*0.15 where empno='E0100';  1 SALARY HAS BEEN UPDATED |

PL/SQL procedure successfully completed.

**//AFTER**

SQL> select empno,salary from employee where empno='E0100';

EMPNO SALARY

---------- ----------

E0100 52900

SQL>

1. **The HRD manager decides to raise the salary of employees working as ‘analyst’ by 0.15. Write a cursor to update the salary of the employees. Display the no. of employee records that has been modified.**

**Query**

**//BEFORE**

SQL> select salary from employee where job='ANALYST';

SALARY

----------

23800

28420

SQL> edit E:\plsql\_ans\cursor\_q2\_HRDanalyst.sql

[[

|  |
| --- |
| declare  begin  update employee set salary=salary+salary\*0.15 where job='ANALYST';  if sql%found then  dbms\_output.put\_line(sql%rowcount||' records are updated');  else  dbms\_output.put\_line('No records are updated');  end if;  end; |

]]

SQL> @ E:\plsql\_ans\cursor\_q2\_HRDanalyst.sql

14 /

2 records are updated

PL/SQL procedure successfully completed.

**//AFTER**

SQL> select salary from employee where job='ANALYST';

SALARY

----------

27370

32683

SQL>

|  |  |
| --- | --- |
| **Description 4.2.2** | **Illustration of Explicit Cursors.** |
| **Date** | 10/09/2023 |

1. **Write an explicit cursor to display the name,department, salary of the first 5 employees getting the highest salary.**

**Query**

SQL> edit E:\plsql\_ans\explicit\_cur\_highestsal.sql****

[[

|  |
| --- |
| declare  cursor empcur is  select empname,deptno,salary from employee order by salary desc;  ename employee.empname%type;  deptno employee.deptno%type;  sal employee.salary%type;  begin  dbms\_output.put\_line('Highest 5 Employee details ');  open empcur;  fetch empcur into ename,deptno,sal;  while empcur%found and empcur%rowcount<=5  loop  dbms\_output.put\_line('Emp Name : '||ename);  dbms\_output.put\_line('Dept No: '||deptno);  dbms\_output.put\_line('Salary : '||sal);  dbms\_output.put\_line('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*');  fetch empcur into ename,deptno,sal;  end loop;  close empcur;  end; |

]]

SQL> @ E:\plsql\_ans\explicit\_cur\_highestsal.sql

27 /

****

PL/SQL procedure successfully completed.

SQL> set serveroutput on;

SQL> @ E:\plsql\_ans\explicit\_cur\_highestsal.sql

27 /

|  |
| --- |
| Highest 5 Employee details  Emp Name : ARNOLD LEONARD AMON  Dept No: A00  Salary : 152750  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Emp Name : DONA ANICE SIBY  Dept No: A00  Salary : 46500  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Emp Name : PHILIP VINCENT  Dept No: B01  Salary : 41250  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Emp Name : ALFRIN LUIZ  Dept No: E01  Salary : 40175  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Emp Name : SHILVY K K  Dept No: C01  Salary : 38250  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |

PL/SQL procedure successfully completed.

1. **The HRD manager decides to raise the salary of employees working as ‘analyst’ by 0.15. Whenever any such raise is given to the employees, a record for the same is maintained in the emp\_raise table. It includes the employee number, the date when the raise was given and actual raise. Write a PL/SQL block to update the salary of the employees and insert a record in the emp\_raise table. Emp\_raise(empcode, raisedate,raise\_amt.)**

**Query**

SQL> edit E:\plsql\_ans\explicit\_cur\_analystraise2.sql

[[

|  |
| --- |
| declare  cursor c\_analyst is  select empno,salary from employee where job='ANALYST';  v\_empno employee.empno%type;  v\_sal employee.salary%type;  v\_raise number;  begin  open c\_analyst;  fetch c\_analyst into v\_empno,v\_sal;  while c\_analyst%found  loop  v\_raise:=v\_sal\*0.15;  update employee set salary=salary+v\_raise where empno=v\_empno;  insert into emp\_raise values(v\_empno,SYSDATE,V\_raise);  fetch c\_analyst into v\_empno,v\_sal;  end loop;  close c\_analyst;  end; |

]]

****

SQL> @ E:\plsql\_ans\explicit\_cur\_analystraise2.sql

29 /

PL/SQL procedure successfully completed.

SQL> select \* from emp\_raise;

EMPCODE RAISEDATE RAISE\_AMT

---------- --------- ----------

E0130 23-SEP-23 3570

E0140 23-SEP-23 4263

****

|  |  |
| --- | --- |
| **Description 4.3** | **Illustration of Procedure.** |
| **Date** | 10/09/2023 |

1. **Write a PL/SQL block which makes use of a stored procedure Proj\_emp ( emp\_name varchar2(50) ) which finds all the details of the projects involved by the given employee.**

**Query**

SQL> edit E:\plsql\_ans\procedure\_illustartion.sql

**//PROCEDURE**

[[

|  |
| --- |
| create or replace procedure proj\_emp(emp\_name IN varchar)  AS  CURSOR projectfinder IS select projname from emp\_proj ep INNER JOIN employee e ON e.empno=ep.empno INNER JOIN project p on ep.projno=p.projno where empname=emp\_name;  project\_name varchar(30);  begin  OPEN projectfinder;  LOOP  FETCH projectfinder INTO project\_name;  EXIT when projectfinder%NOTFOUND;  dbms\_output.put\_line(project\_name);  END LOOP;  CLOSE projectfinder;  end; |

]]

****

SQL> @ E:\plsql\_ans\procedure\_illustartion.sql

22 /

Procedure created.

SQL> edit E:\plsql\_ans\procedure\_illustartion\_main.sql

**//MAIN**

[[

|  |
| --- |
| declare  empname varchar(30);  begin  empname:=&employee\_name;  proj\_emp(empname);  end; |

****

]]

SQL> @ E:\plsql\_ans\procedure\_illustartion\_main.sql

10 /

|  |
| --- |
| Enter value for employee\_name: 'PRIYA TOMY'  old 5: empname:=&employee\_name;  new 5: empname:='PRIYA TOMY';  USER EDUCATION  QUERY SERVICES |

PL/SQL procedure successfully completed.

SQL>

1. **Write a procedure to check whether a string is a palindrome . Call the procedure to list all the palindrome names in the employee table.**

**Query**

SQL> edit E:\plsql\_ans\procedure\_illustartion\_2.sql

**//PROCEDURE**

[[

|  |
| --- |
| CREATE OR REPLACE PROCEDURE palindrome\_checker(arg in varchar2)  AS  rev varchar2(30);  begin  SELECT reverse(arg) INTO rev from DUAL;  if arg = rev  THEN  dbms\_output.put\_line(arg);  end if;  end; |

]]

SQL> @ E:\plsql\_ans\procedure\_illustartion\_2.sql

14 /

Procedure created.

SQL> edit E:\plsql\_ans\procedure\_illustartion\_2\_main.sql

**//MAIN**

[[

|  |
| --- |
| declare  CURSOR empnames IS SELECT UPPER(empname) from employee;  empname varchar(30);  begin  OPEN empnames;  LOOP  FETCH empnames INTO empname;  EXIT WHEN empnames%NOTFOUND;  palindrome\_checker(empname);  END LOOP;  CLOSE empnames;  end; |

]]

SQL> @ E:\plsql\_ans\procedure\_illustartion\_2\_main.sql

19 /

PL/SQL procedure successfully completed.****

SQL> execute palindrome\_checker('MALAYALAM');

MALAYALAM

PL/SQL procedure successfully completed.

SQL> execute palindrome\_checker('MALAYALA');

PL/SQL procedure successfully completed.

1. **Write a PL/SQL block which retrieve all the employee into a cursor and display the details of all assigned projects for each employee using a stored procedure Proj\_emp ( emp\_name varchar2(50).**

**Query**

SQL> edit E:\plsql\_ans\procedure\_illustartion.sql****

**//PROCEDURE**

[[

|  |
| --- |
| create or replace procedure proj\_emp(empname in varchar2) as  cursor emp is select projname,projno from employeee1 e,project p,emp\_projj ep where e.empno=ep.empno and p.projno=ep.projno and empname=empname;  project\_name project.projname%type;  Project\_no project.projno%type;  begin  open emp;  loop  fetch emp into project\_name;  exit when emp%notfound;  dbms\_output.put\_line('project name '||project\_name);  dbms\_output.put\_line('project No '||project\_no);  end loop;  close emp;  end; |

]]

SQL> @ E:\plsql\_ans\procedure\_illustartion.sql

22 /

Procedure created.

SQL> edit E:\plsql\_ans\procedure\_illustartion\_main.sql

**//MAIN**

[[

|  |
| --- |
| Declare  Create emp\_cursor is select empname from employee;  Empname employeee.empname%type;  begin  Open emp\_cursore;  Loop  Fetch emp\_cursor intoempname;  Proj\_name(empname);  End loop;  end;  Declare  Create emp\_cursor is select empname from employee;  Empname employeee.empname%type;  begin  Open emp\_cursore;  Loop  Fetch emp\_cursor intoempname;  Proj\_name(empname);  End loop;  end; |

]]

SQL> @ E:\plsql\_ans\procedure\_illustartion\_main.sql

10 /

Procedure created.

PL/SQL procedure successfully completed.

SQL>

****

========= ARNOLD LEONARD AMON ==========

ADMIN SERVICES

WELD LINE AUTOMATION

-

========= PHILIP VINCENT ==========

ADMIN SERVICES

W L PROGRAM DESIGN

WELD LINE PLANNING

-

========= SHILVY K K ==========

QUERY SERVICES

USER EDUCATION

-

========= ALFRIN LUIZ ==========

OPERATION SUPPORT

GEN SYSTEM SERVICES

-

========= OSHINA ANTONY ==========

ADMIN SERVICES

W L PROGRAMMING

-

========= BINCY PAUL ==========

GENERAL AD SYSTEMS

-

========= ANAMIKA PAUL ==========

GENERAL AD SYSTEMS

OPERATION

-

========= ANEESH DENNY ==========

GENERAL AD SYSTEMS

SYSTEMS SUPPORT

-

========= DONA ANICE SIBY ==========

ACCOUNT PROGRAMMING

-

========= JUNAID K V ==========

W L PROGRAM DESIGN

-

========= CHRISTEENA THOMAS ==========

PERSONNEL PROGRAMMING

ACCOUNT PROGRAMMING

W L PROGRAM DESIGN

-

========= JEFFIN DOMINIC ==========

ADMIN SERVICES

PERSONNEL PROGRAMMING

-

========= JEWEL BIJOY ==========

W L ROBOT DESIGN

GEN SYSTEM SERVICES

SYSTEMS SUPPORT

-

========= MELLOW REEBA THOMAS ==========

GEN SYSTEM SERVICES

-

========= JOHN VARGHESE ==========

-

========= ASHREENA HASSAN ==========

-

========= VISHAK VIJAYAKUMAR ==========

WELD LINE PLANNING

-

========= CORRINE ELIZABETH RODRIGUES ==========

W L ROBOT DESIGN

-

========= MERLIN M.D ==========

W L ROBOT DESIGN

-

========= MARIA JOHN ==========

-

========= VISHALAKSHI V PRABHU ==========

PAYROLL PROGRAMMING

-

========= ANGEL PAUL ==========

PAYROLL PROGRAMMING

-

========= RIYA TONEY ==========

PERSONNEL PROGRAMMING

USER EDUCATION

-

========= PRIYA TOMY ==========

QUERY SERVICES

USER EDUCATION

-

========= ARYAMOL ASOKAN ==========

ACCOUNT PROGRAMMING

USER EDUCATION

-

========= GEO GEORGE ==========

-

========= JIMMY THOMSON ==========

OPERATION

-

========= ALAN PAYYAPPILLY ==========

SCP SYSTEM SUPPORT

-

========= BEN PETER MATHEW ==========

OPERATION SUPPORT

-

========= KRISHNANUNNI S ==========

SCP SYSTEM SUPPORT

-

========= AHALYA V A ==========

WELD LINE AUTOMATION

APPLICATIONS SUPPORT

-

========= ANJALI NAIR ==========

W L PROGRAMMING

-

|  |  |
| --- | --- |
| **Description 4.4** | **Illustration of Functions.** |
| **Date** | 10/09/2023 |

1. **Write a function to find the reverse of EmpNo in Employee table and display the EmpNo and Reversed(Emp No) of the first 5 employees using an SQL Query.**

**Query**

SQL> edit E:\plsql\_ans\function\_illustartion\_1.sql

[[

|  |
| --- |
| CREATE OR REPLACE FUNCTION Reversed(empno in varchar2)  return varchar2  IS  rev varchar2(20);  BEGIN  select reverse(empno) into rev from dual;  return rev;  END; |

****

]]

SQL> @ E:\plsql\_ans\function\_illustartion\_1.sql

9 /

Function created.

SQL> select eno,ename,Reversed(eno) from employee where

rownum<=5;

ENO ENAME REVERSED(ENO)

-------- -------------- ------------------------

E0010 MAJO 0100E

E0011 ABHINAV 1100E

E0012 NIHAL 2100E

E0013 SARA 3100E

E0014 JOHN 4100E

1. **Write a function that would check for the existence of an employee in the employee table given an EmpNo. If existing employee, check whether he is the manager of any department and display messages accordingly.**

**Query**

SQL> edit E:\plsql\_ans\function\_illustartion\_2.sql

[[

|  |
| --- |
| CREATE OR REPLACE FUNCTION CHECK\_emp(empno in varchar2)  return varchar2  AS  j varchar2(20);  BEGIN  SELECT job into j from emp\_tab where eno=empno;  if sql%notfound then  return('EMPLOYEE NOT FOUND');  elsif j='MANAGER' then  return('MANAGER');  else  return('NOT MANAGER');  end if;  END; |

*]]*

SQL> @ E:\plsql\_ans\function\_illustartion\_2.sql

15 /

Function created. ****

SQL> select check\_emp('E0010') from dual;

CHECK\_EMP('E0010')

------------------------

NOT MANAGER

SQL> insert into emp\_tab values ('E0020','SYED',9000, 'MANAGER');

1 row created.

SQL> select check\_emp('E0020') from dual;

CHECK\_EMP('E0020')

-------------------------

MANAGER

|  |  |
| --- | --- |
| **Description 4.5** | **Illustration of Triggers.** |
| **Date** | 10/09/2023 |

1. **Consider the table Employee. Write PL/SQL statements to create a trigger when fired checks the operation performed on a table and based on the operation, a variable is assigned the value ‘update’ or ‘delete’. Previous values of the modified record of the table Employee are stored into the appropriate variables declared and inserted to the audit table AuditEmployee.**

**Query**

SQL> Create table AuditEmployee(eno varchar2(5),ename varchar2(20), esal number(10,2),

job varchar2(20), Audit\_action varchar2(20));

Table created.

SQL> edit E:\plsql\_ans\trigger\_illustartion\_1.sql

[[

|  |
| --- |
| Create or Replace TRIGGER trigaudit BEFORE delete or  update on Emp\_tab  for each row  DECLARE  audit\_action varchar2(20);  BEGIN  if deleting then  audit\_action:='DELETE';  elsif updating then  audit\_action:='UPDATE';  end if;  insert into AuditEmployee values(:old.eno,:old.ename,:old.esal,:old.job,audit\_action);  END; |

]]****

SQL> @ E:\plsql\_ans\trigger\_illustartion\_1.sql

13 /

Trigger created.

SQL> update emp\_tab set esal=10000 where eno='E0013';

1 row updated.

SQL> delete from emp\_tab where eno='E0013';

1 row deleted. ****

SQL> select \* from AuditEmployee;

ENO ENAME ESAL JOB AUDIT\_ACTION

--------- --------- --------- ---------- ------------

E0013 SARA 17000 PROFESSOR DELETE

E0013 SARA 17000 PROFESSOR UPDATE

1. **Write PL/SQL statements to create a trigger which generates an error messages if the salary is below or beyond the valid range 0-5000 on the employee table. The triggering events are update and insert.**

**Query**

SQL> edit E:\plsql\_ans\trigger\_illustartion\_2.sql

[[

|  |
| --- |
| *create or replace trigger Trig2 after INSERT OR UPDATE*  *ON emp\_tab*  *for each row*  *begin*  *if (:new.esal not between 0 and 5000 ) then*  *RAISE\_APPLICATION\_ERROR(-20500,'SALARY RANGE NOT*  *BETWEEN 0 AND 5000');*  *END IF;*  *END;* |

*]]*

SQL> @ E:\plsql\_ans\trigger\_illustartion\_2.sql

8 /

Trigger created.

****

SQL> update emp\_tab set esal=6000 where eno='E0010';

update emp\_tab set esal=6000 where eno='E0010'

\*

ERROR at line 1:

ORA-20500: SALARY RANGE NOT BETWEEN 0 AND 5000

ORA-06512: at "MCA.TRIG2", line 3

ORA-04088: error during execution of trigger

'MCA.TRIG2'

1. **Write PL/SQL statements to create a trigger that limits the DML actions to the Employee table to weekdays from 8.30am to 6.30pm. If a user tries to insert/update/delete a row in the Employee table, a warning message will be prompted.**

**Query**

SQL> edit E:\plsql\_ans\trigger\_illustartion\_3.sql

[[

|  |
| --- |
| create or replace trigger mytrig1 BEFORE DELETE OR  INSERT OR UPDATE ON emp\_tab  begin  if (to\_char(sysdate,'day') in ('sun','mon')) or  (to\_char(sysdate ,'hh:mi') not between '08:30' and  '18:30') then  RAISE\_APPLICATION\_ERROR(-20500,'TABLE IS SECURED');  END IF;  END; |

]]

SQL> @ C:\Users\cclab29\OneDrive\mca2333\p25.sql

7 /

Trigger created.

SQL> insert into emp\_tab values('E0013','SARA',17000,'PROFESSOR');

insert into emp\_tab values('E0013','SARA',17000,'PROFESSOR')

\*

ERROR at line 1:

ORA-20500: TABLE IS SECURED

ORA-06512: at "SYSTEM.UP\_TRIG", line 3

ORA-04088: error during execution of trigger

'SYSTEM.UP\_TRIG'