



الجمهورية العربية السورية

جامعة دمشق

كلية الهندسة المعلوماتية

## تقرير إنجاز مشروع قواعد معطيات متقدمة

إعداد الطلاب:

مرغد خالد خليفة

مجد مروان الخضر

عبد الله محمد المقداد

حكم محمد واجد جبين

محمود رسول مشلح

اختصاص: برمجيات

إشراف: م. عبد البديع مراد

م. أبو الخير الصوص

## القسم الأول

### أولاً:

١- تعريف TableSpace باسم homeworkts حجمه MB400 مؤلف من أربعة ملفات معطيات datafile .

```
CREATE TABLESPACE HOMEWORKTS
DATAFILE 'C:\app\Ali\oradata\orcl\HOMEWORKTSDF01.DBF' SIZE 100M,
         'C:\app\Ali\oradata\orcl\HOMEWORKTSDF02.DBF' SIZE 100M,
         'C:\app\Ali\oradata\orcl\HOMEWORKTSDF03.DBF' SIZE 100M,
         'C:\app\Ali\oradata\orcl\HOMEWORKTSDF04.DBF' SIZE 100M
LOGGING
EXTENT MANAGEMENT LOCAL
SEGMENT SPACE MANAGEMENT AUTO;
```

Query Result x Script Output x

Task completed in 2.486 seconds

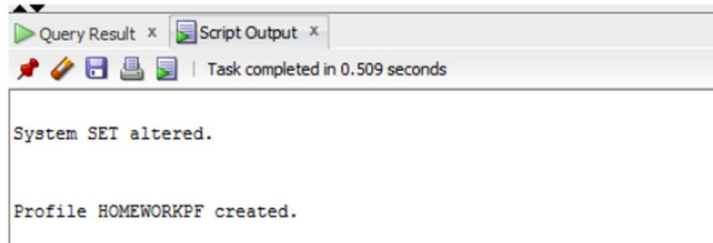
User HR altered.

TABLESPACE HOMEWORKTS created.

٢- تعريف Profile باسم homeworkpf يحدد من خلاله السماح لمستخدم واحد الاتصال بقاعدة المعطيات بشرط أن تكون مدة الاتصال الفعال ساعة ومدة الاتصال الغير فعالة عشرة دقائق وضرورة تغيير كلمة السر كل سبعة أيام وحجم الذاكرة ٥٠ كيلو بايت.

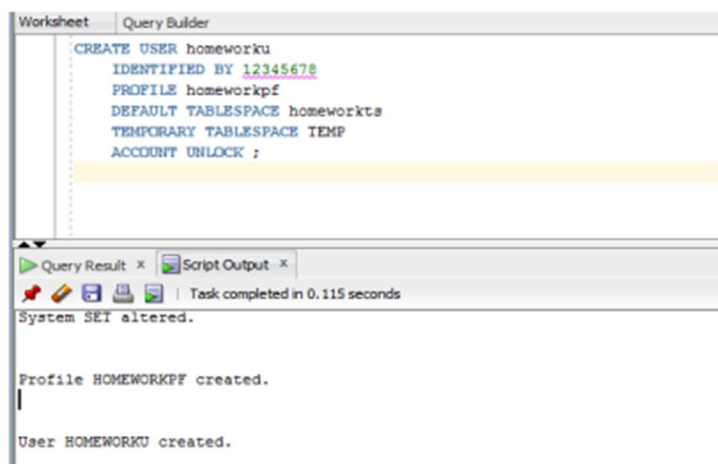
```
ALTER SYSTEM SET RESOURCE_LIMIT = TRUE ;
```

```
CREATE PROFILE homeworkpf LIMIT  
IDLE_TIME 10  
CONNECT_TIME 60  
SESSIONS_PER_USER 1  
PASSWORD_LIFE_TIME 7  
PRIVATE_SGA 50
```



٣- تعريف حساب User جديد في قاعدة المعطيات باسم homeworkku مرتبط ب-  
TableSpace

باسم homeworkkts ومنحه profile homeworkkpf.



```
CREATE USER homeworkku
IDENTIFIED BY 12345678
PROFILE homeworkkpf
DEFAULT TABLESPACE homeworkkts
TEMPORARY TABLESPACE TEMP
ACCOUNT UNLOCK ;
```

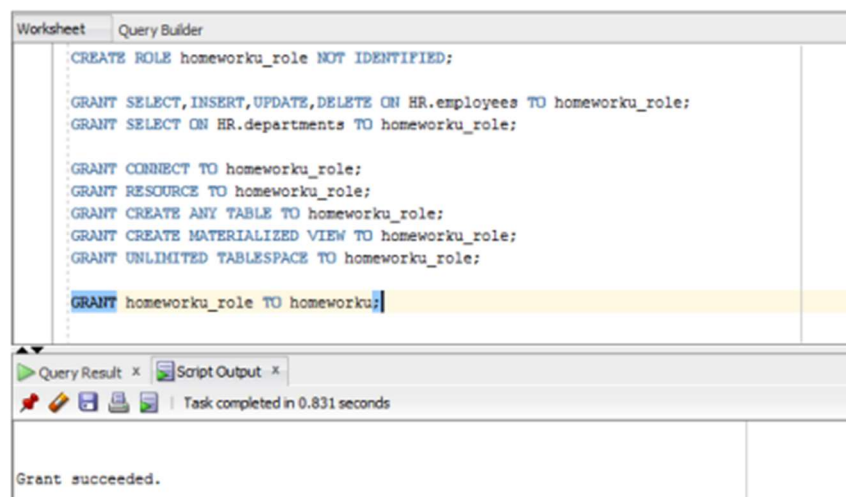
Task completed in 0.115 seconds

System SET altered.

Profile HOMEWORKKPF created.

User HOMEWORKKU created.

٤- إعطاء كافة الصلاحيات لهذا المستخدم من خلال Roles واحدة، وأن يكون له  
صلاحيات قراءة من جدول departments وتعديل وحذف بيانات جدول  
employees من حساب HR.



```
CREATE ROLE homeworkku_role NOT IDENTIFIED;

GRANT SELECT, INSERT, UPDATE, DELETE ON HR.employees TO homeworkku_role;
GRANT SELECT ON HR.departments TO homeworkku_role;

GRANT CONNECT TO homeworkku_role;
GRANT RESOURCE TO homeworkku_role;
GRANT CREATE ANY TABLE TO homeworkku_role;
GRANT CREATE MATERIALIZED VIEW TO homeworkku_role;
GRANT UNLIMITED TABLESPACE TO homeworkku_role;

GRANT homeworkku_role TO homeworkku;
```

Task completed in 0.831 seconds

Grant succeeded.

٥ - إجراء نسخة احتياطية للحساب homeworkku دون أخذ بيانات الجداول ،  
وعرض ملف LOG

```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Ali> EXP.EXE USERID='sys/Passw0rd as sysdba'

Export: Release 11.2.0.2.0 - Production on Mon Dec 4 19:54:40 2023
Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Connected to: Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - 64bit
Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
Export done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set

About to export specified users ...
. exporting pre-schema procedural objects and actions
. exporting foreign function library names for user SYS
. exporting PUBLIC type synonyms
. exporting private type synonyms
. exporting object type definitions for user SYS
About to export SYS's objects ...
. exporting database links
. exporting sequence numbers
. exporting cluster definitions
. exporting synonyms
. exporting views
EXP-00104: datatype <BINARY_DOUBLE> of column CALLSPERSEC in table SYS.U_$VLM_PC
METRIC is not supported, table will not be exported
EXP-00104: datatype <BINARY_DOUBLE> of column RESPONSETIMEPERCALL in table SYS.U
_$VLM_PCMETRIC_HISTORY is not supported, table will not be exported
EXP-00104: datatype <BINARY_DOUBLE> of column CALLSPERSEC in table SYS.GU_$VLM_P
CMETRIC is not supported, table will not be exported
EXP-00104: datatype <BINARY_DOUBLE> of column RESPONSETIMEPERCALL in table SYS.G
U_$VLM_PCMETRIC_HISTORY is not supported, table will not be exported
. exporting stored procedures
. exporting operators
. exporting referential integrity constraints
. exporting triggers
. exporting indextypes

```

## القسم الثاني:

### قواعد المعالجة PL-SQL:

باستخدام لغة PL-SQL القواعد والإجراءات التي تحقق ما يلي:

1. إضافة الأفلام تتم فقط يوم الخميس بين الساعة السادسة والثامنة صباحا،  
ويمنع حذف أي فيلم.

#### deletion films not allowed trigger .

```
CREATE OR REPLACE TRIGGER Film_Avoid_Deletion
BEFORE DELETE ON homeworkku.Film
BEGIN
    RAISE_APPLICATION_ERROR(-20001, 'Deletion of films is not allowed.');
```

Script Output x

Task completed in 0.907 seconds

Trigger FILM\_AVOID\_DELETION compiled

#### delete film no allowed trigger result.

```
DELETE FROM film WHERE ID = 1;
```

Script Output x Query Result x

Task completed in 0.171 seconds

Error starting at line : 6 in command -  
DELETE FROM film WHERE ID = 1  
Error report -  
ORA-20001: Deletion of films is not allowed.  
ORA-06512: at "HOMEWORKKU.FILM\_AVOID\_DELETION", line 2  
ORA-04088: error during execution of trigger 'HOMEWORKKU.FILM\_AVOID\_DELETION'

## 1-2 Film\_Limit\_Addition trigger.

```
CREATE OR REPLACE TRIGGER Film_Limit_Addition
BEFORE INSERT ON homeworks.Film
DECLARE
    today_day VARCHAR2(10);
BEGIN
    SELECT TO_CHAR(SYSDATE, 'DY') INTO today_day FROM DUAL;
    IF today_day = 'THU' THEN
        IF TO_CHAR(SYSDATE, 'HH24:MI') NOT BETWEEN '06:00' AND '08:00' THEN
            RAISE_APPLICATION_ERROR(-20002, 'Film additions are allowed only on Thursdays between 6:00 AM and 8:00 AM.');
```

Script Output x

Task completed in 0.596 seconds

Trigger FILM\_LIMIT\_ADDITION compiled

## Film\_Limit\_Addition trigger result.

```
INSERT INTO "homeworks"."Film" (ID, TITLE, DESCRIPTION, RELEASE_YEAR, RENTAL_DURATION, RENTAL_RATE, LENGTH, REPLACEMENT_COST, RATING, LAST_UPDATE,
VALUES ('2', 'F', 'F', '2', 123, '432', '234', '324', TO_TIMESTAMP('2023-12-06 22:40:41.447000000', 'YYYY-MM-DD HH24:MI:SS.FF'), '4', '453', '1')
```

Script Output x

Task completed in 0.06 seconds

INSERT INTO "homeworks"."Film" (ID, TITLE, DESCRIPTION, RELEASE\_YEAR, RENTAL\_DURATION, RENTAL\_RATE, LENGTH, REPLACEMENT\_COST, RATING, LAST\_UPDATE, SPECI  
VALUES ('2', 'F', 'F', '2', 123, '432', '234', '324', TO\_TIMESTAMP('2023-12-06 22:40:41.447000000', 'YYYY-MM-DD HH24:MI:SS.FF'), '4', '453', '1')

Error report -

ORA-20002: Film additions are allowed only on Thursdays.

ORA-04512: at "SYSTEM.FILM\_LIMIT\_ADDITION", line 10

ORA-04512: error during execution of trigger "SYSTEM.FILM\_LIMIT\_ADDITION"

٢. قيمة الإيجار في جدول الدفعات يجب أن تكون مساوية لقيمة تكلفة الإيجار من جدول الأفلام كافة أيام الأسبوع، إلا في حال كان يوم الإيجار سبت أو أحد تضاف قيمة ١٥٪ على تكلفة الإيجار.

```
CREATE OR REPLACE TRIGGER update_payment_amount
BEFORE INSERT OR UPDATE ON Payment
FOR EACH ROW
DECLARE
    v_rental_cost NUMBER;
    v_rental_date DATE;
    v_rental_day VARCHAR(9);
BEGIN
    SELECT f.Rental_Rate, r.Rental_Date
    INTO v_rental_cost, v_rental_date
    FROM Film f
    JOIN Inventory inv ON f.ID = inv.FilmID
    JOIN Rental r ON inv.ID = r.InventoryID
    AND ROWNUM = 1
    ORDER BY r.Rental_Date DESC;

    v_rental_day := TO_CHAR(v_rental_date, 'DAY');

    IF v_rental_day IN ('SATURDAY', 'SUNDAY') THEN
        :NEW.Amount := v_rental_cost * 1.15;
    ELSE
        :NEW.Amount := v_rental_cost;
    END IF;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        NULL;
    WHEN OTHERS THEN
        NULL;
END;
```

٣. منع تأجير نفس الفيلم لنفس الزبون أكثر من مرة بالشهر.

```
CREATE OR REPLACE TRIGGER prevent_same_film_rental
BEFORE INSERT ON Rental
FOR EACH ROW
DECLARE
    v_rental_count NUMBER;
BEGIN
    SELECT COUNT(*)
    INTO v_rental_count
    FROM Rental r
    JOIN Inventory inv ON r.InventoryID = inv.ID
    JOIN Film f ON inv.FilmID = f.ID
    WHERE r.CustomerID = :NEW.CustomerID
    AND inv.ID = :NEW.InventoryID
    AND EXTRACT(MONTH FROM r.Rental_Date) = EXTRACT(MONTH FROM SYSDATE)
    AND EXTRACT(YEAR FROM r.Rental_Date) = EXTRACT(YEAR FROM SYSDATE);

    IF v_rental_count > 0 THEN
        RAISE_APPLICATION_ERROR(-20001, 'غير مسموح بتأجير نفس الفيلم لنفس الزبون أكثر من مرة في الشهر');
    END IF;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        NULL;
    WHEN OTHERS THEN
        NULL;
END;
```



٤. بناء سجل متابعة لعمليات الإضافة والتعديل لجداول الحجوزات في جدول رديف (يجب بناء الجدول)
- يسجل قيم جميع الحقول قبل وبعد تنفيذ العملية مع نوع وتاريخ وزمن العملية .

#### 4- audit table .

```

CREATE TABLE homeworkku.Rental_Audit (
  Audit_ID NUMBER ,
  Rental_ID INT NOT NULL,
  Action_Type VARCHAR2(10) NOT NULL,
  StaffID INT NOT NULL,
  CustomerID INT NOT NULL,
  InventoryID INT NOT NULL,
  Rental_Date_before TIMESTAMP DEFAULT NULL,
  Rental_Date_after TIMESTAMP DEFAULT NULL,
  Return_Date_before TIMESTAMP DEFAULT NULL,
  Return_Date_after TIMESTAMP DEFAULT NULL,
  Last_Update_before TIMESTAMP DEFAULT NULL,
  Last_Update_after TIMESTAMP DEFAULT NULL,
);

```

Query Result x Script Output x

Task completed in 0.071 seconds

02000. 00000 - "missing \$\$ keyword"

Table HOMEWORKKU.RENTAL\_AUDIT created.

#### 4- create sequence and table for audit on rentals table.

```

/* 4 */
CREATE TABLE homeworkku.Rental_Audit (
  Audit_ID NUMBER GENERATED ALWAYS AS IDENTITY,
  Rental_ID INT NOT NULL,
  Action_Type VARCHAR2(10) NOT NULL,
  StaffID INT NOT NULL,
  CustomerID INT NOT NULL,
  InventoryID INT NOT NULL,
  Rental_Date_before TIMESTAMP DEFAULT NULL,
  Rental_Date_after TIMESTAMP DEFAULT NULL,
  Return_Date_before TIMESTAMP DEFAULT NULL,
  Return_Date_after TIMESTAMP DEFAULT NULL,
  Last_Update_before TIMESTAMP DEFAULT NULL,
  Last_Update_after TIMESTAMP DEFAULT NULL,
  Action_Timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

DROP SEQUENCE auditseq;

CREATE SEQUENCE auditseq
  MINVALUE 1
  MAXVALUE 999999
  INCREMENT BY 1
  START WITH 1
  NOCACHE
  ORDER
  NOCYCLE;

```

#### 4- triggers for insert and update rows on rental table for audit.

```
CREATE OR REPLACE TRIGGER Rental_Insert_Audit_Trigger
AFTER INSERT ON homeworkku.Rental
FOR EACH ROW
DECLARE
    v_audit_seq NUMBER;
BEGIN
    SELECT homeworkku.auditseq.nextval INTO v_audit_seq FROM dual;

    INSERT INTO homeworkku.Rental_Audit (Audit_ID,Rental_ID,Action_Type,StaffID,CustomerID,InventoryID,
    Rental_Date_before,Rental_Date_after,Return_Date_before,Return_Date_after,Last_Update_before,Last_Update_after
    )
    VALUES (v_audit_seq,:NEW.ID,'INSERT',:NEW.StaffID,:NEW.CustomerID,:NEW.InventoryID,NEW.Rental_Date,NULL,:NEW.Return_Date,NULL,:NEW.Last_Update
    );
END;

CREATE OR REPLACE TRIGGER Rental_Update_Audit_Trigger
AFTER UPDATE ON homeworkku.Rental
FOR EACH ROW
DECLARE
    v_audit_seq NUMBER;
BEGIN
    SELECT homeworkku.auditseq.nextval INTO v_audit_seq FROM dual;

    INSERT INTO homeworkku.Rental_Audit (Audit_ID,Rental_ID,Action_Type,StaffID,CustomerID,InventoryID,
    Rental_Date_before,Rental_Date_after,Return_Date_before,Return_Date_after,Last_Update_before,Last_Update_after
    )
    VALUES (v_audit_seq,:NEW.ID,'UPDATE',:NEW.StaffID,:NEW.CustomerID,:NEW.InventoryID,:OLD.Rental_Date,:NEW.Rental_Date,
    :OLD.Return_Date,:NEW.Return_Date,:OLD.Last_Update,:NEW.Last_Update
    );
END;
/
```

٥. تسجيل زمن دخول وخروج أي مستخدم لقاعدة المعطيات في جدول مناسب ضمن حساب homeworkku يستثنى من ذلك حسابي Sys, System .





#### 5- audit login result.

Worksheet

Query Builder

```
select * from audit_login;
```

Query Result x

 SQL | All Rows Fetched: 1 in 0.007 seconds

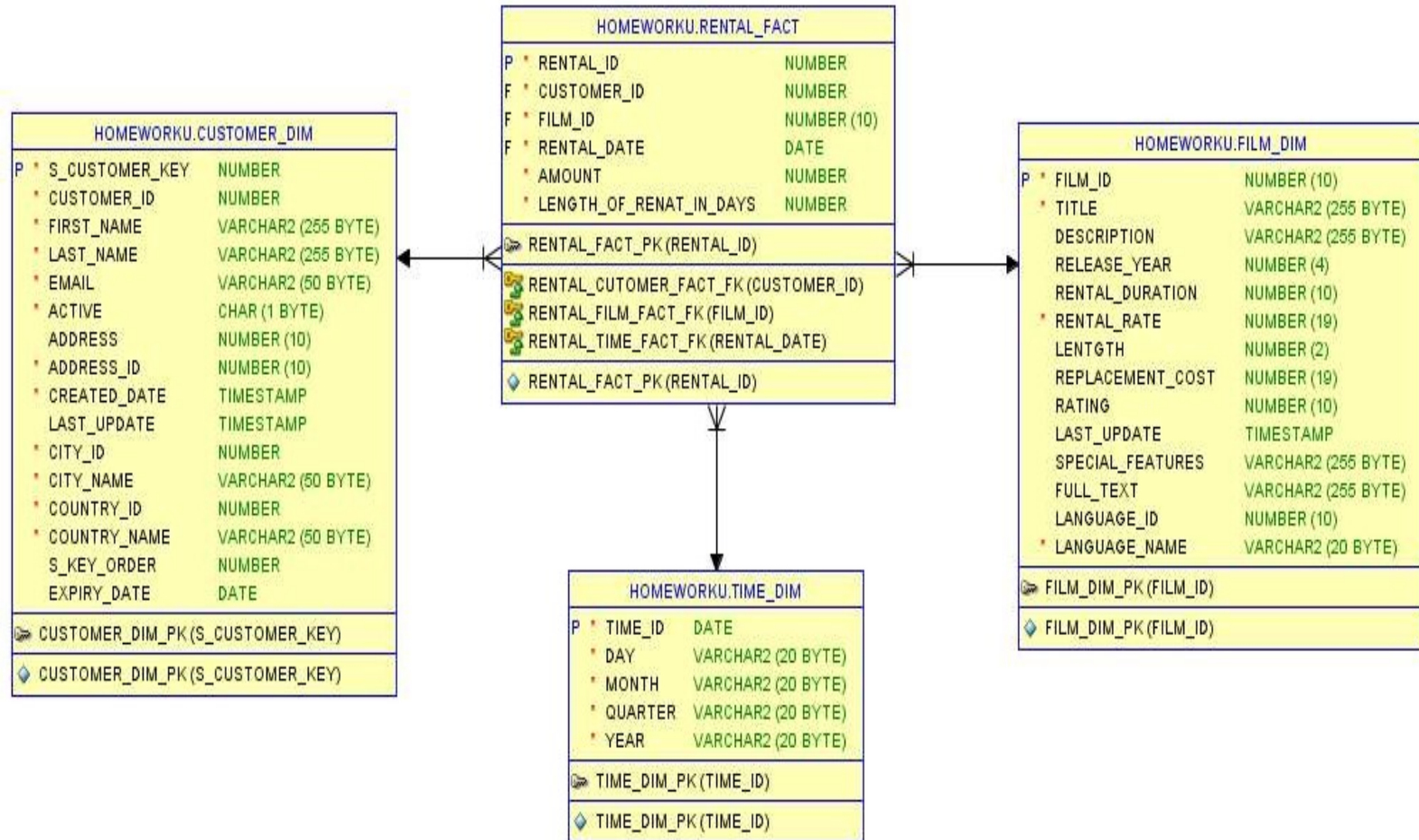
ACCOUNT_NAME	CONNECT_DATE
1 HOMEWORKU	2023:12:10-21:38:54

#### 5- audit login.

Worksheet		Query Builder	
		<pre> BEGIN IF user != 'System' and user != 'Sys' THEN   INSERT INTO audit_login ( account_name, connect_date)   VALUES (user , TO_CHAR(SYSDATE,'YYYY:MM-DD-RH24:MI:SS')); END IF; END;</pre>	
		Script Output x	
		Task completed in 0.39 seconds	
		Table AUDIT_LOGIN created.	
		Trigger LOGONDS_TRIGGER compiled	

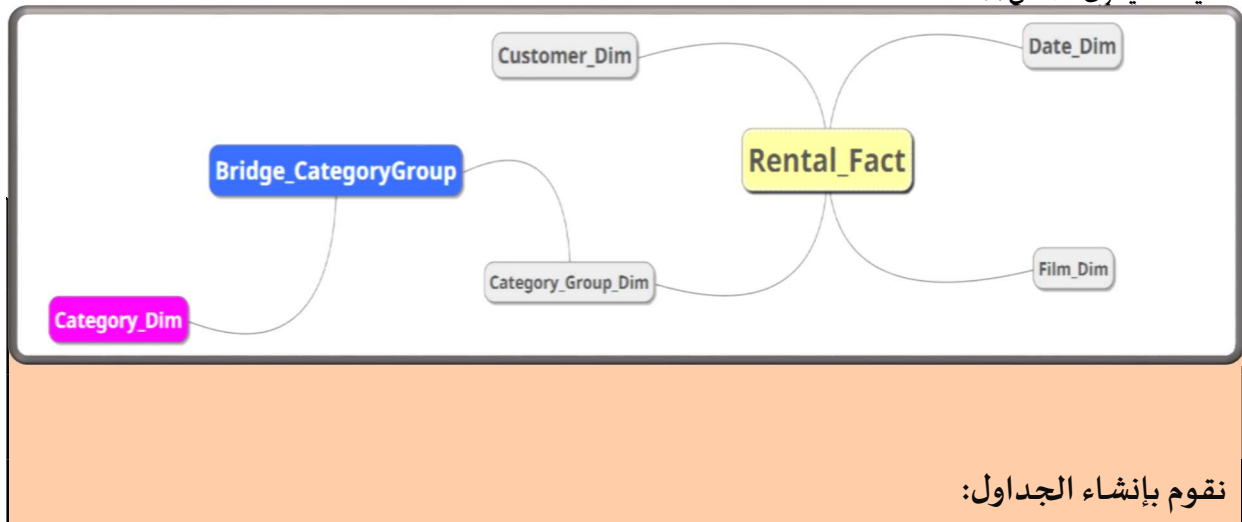
## القسم الثالث: قواعد المعطيات متعددة الابعاد

سنقوم ببناء المخطط من نوع **Star Schema** والذي سيمكننا من الحصول على أعلى أداء في الاستعلامات عن الاحصائيات ومحور ال Fact table هما جدول ال Rental And Payment والمخطط بالشكل التالي:



## ملاحظة مهمة لسبب اختيار المخطط والابعاد:

لم نقم بتضمين ابعاد نحن لسنا بحاجة لها من قراءتنا لنص الاسئلة في هذا القسم مثل ال Staff, Store, etc. ومثلا ان كنا بحاجة لمعرفة فئات الأفلام Category وهي تملك علاقة Many-to-Many مع ال Film هنا سنضطر لجعل المخطط Hybrid أي إضافة تفاصيل للأبعاد لن يتم البحث ضمنها الا عند الحاجة لتفاصيل فئات الفلم مثلا ونقوم على سبيل المثال باستخدام طريقة ال Bridge Table كإحدى الطرق لحل هذه المشكلة والشكل التالي هو مجرد شكل لتوضيحي كيف سيكون الشكل لمخطط:



### Film\_Dim

```
CREATE TABLE film_dim (
  film_id          NUMBER NOT NULL,
  title            VARCHAR2(255) NOT NULL,
  description       VARCHAR2(255),
  release_year     NUMBER(4),
  rental_duration  NUMBER(10),
  rental_rate      NUMBER(19, 0) NOT NULL,
  lentgth         NUMBER(2),
  replacement_cost NUMBER(19, 0),
  rating          NUMBER(10),
  last_update      TIMESTAMP,
  special_features VARCHAR2(255),
  full_text        VARCHAR2(255),
  language_id      NUMBER(10),
  language_name    VARCHAR2(20) NOT NULL,
  CONSTRAINT film_dim_pk PRIMARY KEY (film_id)
);
```

### Customer\_Dim

```
CREATE TABLE customer_dim (
  s_customer_key  NUMBER NOT NULL,
  customer_id     NUMBER NOT NULL,
  first_name      VARCHAR2(255) NOT NULL,
  last_name       VARCHAR2(255) NOT NULL,
  email           VARCHAR2(50) NOT NULL,
  active          CHAR(1) NOT NULL,
  created_date    TIMESTAMP NOT NULL,
  last_update     TIMESTAMP,
  address_id      NUMBER(10) NOT NULL,
  address         VARCHAR2(50) NOT NULL,
  city_id         NUMBER(10) NOT NULL,
  city_name       VARCHAR2(50) NOT NULL,
  country_id      NUMBER(10) NOT NULL,
  country_name    VARCHAR2(50) NOT NULL,
  s_key_order     NUMBER,
  expiry_date     DATE,
  CONSTRAINT customer_dim_pk PRIMARY KEY
    ( s_customer_key )
);
```

## Time\_Dim

```
CREATE TABLE time_dim (  
  time_id DATE NOT NULL,  
  day VARCHAR2(20) NOT NULL,  
  month VARCHAR2(20) NOT NULL,  
  quarter VARCHAR2(20) NOT NULL,  
  year VARCHAR2(20) NOT NULL,  
  CONSTRAINT time_dim_pk PRIMARY KEY (time_id)  
);
```

## Rental\_Fact

```
CREATE TABLE rental_fact (  
  rental_id NUMBER NOT NULL,  
  customer_id NUMBER NOT NULL,  
  film_id NUMBER NOT NULL,  
  rental_date DATE NOT NULL,  
  amount NUMBER NOT NULL,  
  length_of_renat_in_days NUMBER NOT NULL,  
  CONSTRAINT rental_fact_pk PRIMARY KEY (  
    rental_id ),  
  CONSTRAINT rental_customer_fact_fk FOREIGN KEY  
    ( customer_id ) REFERENCES customer_dim  
    ( s_customer_key ),  
  CONSTRAINT rental_film_fact_fk FOREIGN KEY  
    ( film_id ) REFERENCES film_dim ( film_id ),  
  CONSTRAINT rental_time_fact_fk FOREIGN KEY  
    ( rental_date ) REFERENCES time_dim ( time_id )  
);
```

**الطلب الأول: بناء كافة الابعاد مع مراعاة عملية التعديل التاريخية لبعد الزبون، وتحديد جميع الهرميات المناسبة لكل بعد.**

قمنا بإضافة Surrogate Key لجدول الزبون بالإضافة لحقلين إضافيين هما s\_key\_order و expiry\_date وهذه الحقول سنقوم بحفظ جميع التعديلات التي ستتم للزبون. سيتم ضافة حقل جديد عند أي تغيير في بيانات الزبون وستصبح قيم كل من s\_key\_order بقيمة ترتيب التعديل الذي طرأ أي ان كانت المرة الأولى فسيصبح 1 وللمرة الثانية 2 وهكذا وبالنسبة لـ expiry\_date ستصبح بتاريخ حدوث التعديل وهذان الحقلان سيكونان بقيمة null لأخر نسخة لبيانات الزبون وبالنسبة لـ key يوجد عدة طرق لتعبئته وهو ليس مرتبط بأي قيمة من الجدول الأساسي يمكن توليد قيمه من عداد Sequence وملاحظة بهذه الطريقة المستخدمة سيترتب عبء علينا بالاستعلامات التي تخص الزبون لان الزبون قد يوجد له اكثر من حقل في بعد الزبون.

### Film\_Dim

```
CREATE DIMENSION film_dim
  LEVEL film_id IS film_dim.film_id
  LEVEL title IS film_dim.title
  LEVEL description IS film_dim.description
  LEVEL release_year IS
film_dim.release_year
  LEVEL rental_duration IS
film_dim.rental_duration
  LEVEL rental_rate IS film_dim.rental_rate
  LEVEL lentgth IS film_dim.lentgth
  LEVEL replacement_cost IS
film_dim.replacement_cost
  LEVEL rating IS film_dim.rating
  LEVEL last_update IS film_dim.last_update
  LEVEL special_features IS
film_dim.special_features
  LEVEL full_text IS film_dim.full_text
  LEVEL language_id IS film_dim.language_id
  LEVEL language_name IS
film_dim.language_name
  HIERARCHY hierarchy_film ( film_id
    CHILD OF language_id
  )
  ATTRIBUTE language_id DETERMINES (
    film_dim.language_name
  );
```

### Customer\_Dim

```
CREATE DIMENSION customer_dim
  LEVEL s_customer_key IS
customer_dim.s_customer_key
  LEVEL customer_id IS customer_dim.customer_id
  LEVEL first_name IS customer_dim.first_name
  LEVEL last_name IS customer_dim.last_name
  LEVEL email IS customer_dim.email
  LEVEL active IS customer_dim.active
  LEVEL created_date IS customer_dim.created_date
  LEVEL last_update IS customer_dim.last_update
  LEVEL address_id IS customer_dim.address_id
  LEVEL address IS customer_dim.address
  LEVEL city_id IS customer_dim.city_id
  LEVEL city_name IS customer_dim.city_name
  LEVEL country_id IS customer_dim.country_id
  LEVEL country_name IS
customer_dim.country_name
  LEVEL s_key_order IS customer_dim.s_key_order
  LEVEL expiry_date IS customer_dim.expiry_date
  HIERARCHY hierarchy_customer (s_customer_key
    CHILD OF address_id CHILD OF city_id CHILD
OF country_id)
  ATTRIBUTE address_id DETERMINES
(customer_dim.address)
  ATTRIBUTE city_id DETERMINES
(customer_dim.city_name)
  ATTRIBUTE country_id DETERMINES
(customer_dim.country_name);
```

### Time\_Dim

```
CREATE DIMENSION time_dim
  LEVEL time_id IS time_dim.time_id
  LEVEL day IS time_dim.day
  LEVEL month IS time_dim.month
  LEVEL quarter IS time_dim.quarter
  LEVEL year IS time_dim.year
  HIERARCHY hierarchy_time_all (day CHILD OF month CHILD OF quarter CHILD OF year)
  HIERARCHY hierarchy_time (day CHILD OF month CHILD OF year);
```



نقوم بالتحقق من وجود الابعاد بالتعليمة التالية:

```
SELECT * FROM all_dimensions;
```

cript Output x Query Result x

SQL | All Rows Fetched: 3 in 0.008 seconds

	OWNER	DIMENSION_NAME	INVALID	COMPILE_STATE	REVISION
1	HOMEWORKU	FILM_DIM	N	VALID	1
2	HOMEWORKU	TIME_DIM	N	VALID	1
3	HOMEWORKU	CUSTOMER_DIM	N	VALID	1

الطلب الثاني: تجزئة جدول Fact بالطريقة التي تراها مناسبة وتعليل ذلك، هل من الممكن أن تكون هذه التجزئة مركبة وضح ذلك.

نقوم بعمل Drop لجدول ال Fact بحالتنا لأنه خالي من اية سجلات ثم نقوم ببنائه من جديد و Partition معين لكن ان كان يحوي سجلات فيجب عمل تصدير لها ثم Drop للجدول وانشائه بالتقسيم المرادة من جديد وعمل استيراد للبيانات وذلك لعدم القدرة على انشاء اقسام (Partitions) للجدول بعد انشائه (وذلك لجميع نسخ oracle التي دون (12c).

نستخدم **PARTITION BY RANGE** على جدول ال Fact وذلك للحقل rental\_date كل سنة لحال ويمكن التقسيم كل شهر او او .. الخ ولكن سنعتمد على كل سنة بقسم لحال نظرا الى نوع البيانات لدينا فهي حجوزات أفلام فمن المنطق انه مهما بلغ عدد الحجوزات كبير لن تصل الى ارقام ضمن السنة الواحدة ستبطئ عمل الاستعلامات بالشكل التالي:

```
CREATE TABLE rental_fact (
  rental_id      NUMBER NOT NULL,
  customer_id    NUMBER NOT NULL,
  film_id        NUMBER NOT NULL,
  rental_date     DATE NOT NULL,
  amount         NUMBER NOT NULL,
  length_of_renat_in_days NUMBER NOT NULL,
  CONSTRAINT rental_fact_pk PRIMARY KEY ( rental_id ),
  CONSTRAINT rental_cutomer_fact_fk FOREIGN KEY ( customer_id )
REFERENCES customer_dim (s_customer_key ),
  CONSTRAINT rental_film_fact_fk FOREIGN KEY (film_id ) REFERENCES
film_dim (film_id ),
```



```

CONSTRAINT rental_time_fact_fk FOREIGN KEY (rental_date )
REFERENCES time_dim (time_id )
)
PARTITION BY RANGE (rental_date)
(
    PARTITION rentals_2020 VALUES LESS
        THAN(TO_DATE('01/01/2020','DD/MM/YYYY')),
    PARTITION rentals_2021 VALUES LESS
        THAN(TO_DATE('01/01/2021','DD/MM/YYYY')),
    PARTITION rentals_2022 VALUES LESS
        THAN(TO_DATE('01/01/2022','DD/MM/YYYY')),
    PARTITION rentals_2023 VALUES LESS
        THAN(TO_DATE('01/01/2023','DD/MM/YYYY')),
    PARTITION rentals_after_2023 VALUES LESS THAN(MAXVALUE)
);

```

يمكن عمل تجزئة مركبة بمثلنا عن طريق **RANGE - HASH** حيث من الممكن ان نرى ان تجزئة كل سنة قد تحوي الكثير من البيانات فيمكن إضافة hash وسيكون لحقل rental\_id ولكن التقسيمات للجدول ترى من حجم العمل والتوقعات لل Business.

سنستعرض الأقسام للجدول التي تم انشائها:

	PARTITION_NAME	LAST_ANALYZED	NUM_ROWS	BLOCKS	SAMPLE_SIZE	HIGH_VALUE
1	RENTALS_2020	(null)	(null)	(null)	(null)	TO_DATE(' 2020-01-01 00:00:00',
2	RENTALS_2021	(null)	(null)	(null)	(null)	TO_DATE(' 2021-01-01 00:00:00',
3	RENTALS_2022	(null)	(null)	(null)	(null)	TO_DATE(' 2022-01-01 00:00:00',
4	RENTALS_2023	(null)	(null)	(null)	(null)	TO_DATE(' 2023-01-01 00:00:00',
5	RENTALS_AFTER_2023	(null)	(null)	(null)	(null)	MAXVALUE

الطلب الثالث: بناء **MATERIALIZED VIEW** يتضمن اسم الفيلم واسم الزبون وفترة الحجز،  
تحدث بيانات هذا المنظور بشكل تراكمي.

نقوم بإنشاء MV LOG أولاً لكل الجداول التي سنأخذ منها البيانات وهي rental\_fact و film\_dim و customer\_dim بالشكل التالي:

```

CREATE MATERIALIZED VIEW LOG ON customer_dim WITH

```

```

PRIMARY KEY,
ROWID
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW LOG ON film_dim WITH
PRIMARY KEY,
ROWID
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW LOG ON rental_fact WITH
PRIMARY KEY,
ROWID
INCLUDING NEW VALUES;

```

وسبب حاجتنا لهذه الـ LOG VIEW بسبب ماتم طلبه بالسؤال من التحديث بشكل تراكمي فكل فترة سنقوم بعمل refresh ليتم تحديث الـ MATERIALIZED VIEW التي سننشئها تالياً:

```

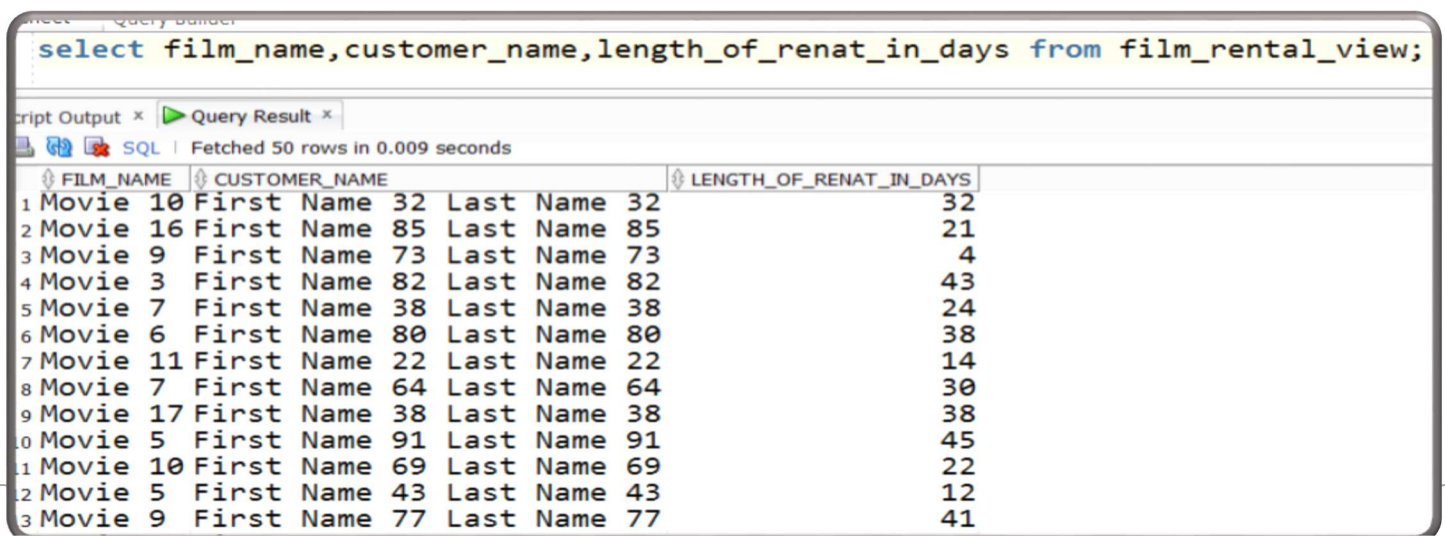
CREATE MATERIALIZED VIEW film_rental_view
BUILD IMMEDIATE REFRESH FAST ON DEMAND
AS
SELECT
    f.rowid film rid, c.rowid custoemr rid, r.rowid rental rid,
    f.title          film_name,
    ( c.first_name || ' ' || c.last_name ) customer_name,
    r.length_of_renat_in_days
FROM
    film_dim f, customer_dim c, rental_fact r
WHERE
    r.customer_id = c.s_customer_key
    AND r.film_id = f.film_id;

```

وضعنا الـ rowid لأنها إجبارية اذا كنا سنقوم بعمل MV من نوع REFRESH FAST والـ MV لا يحوي الـ JOINS بدون

اية توابع تجميعية ([https://docs.oracle.com/en/database/oracle/oracle-database/19/dwhsg/basic-](https://docs.oracle.com/en/database/oracle/oracle-database/19/dwhsg/basic-materialized-views.html%23GUID-8823A06E-853B-4876-AB9F-96D2D4E5A1DE)  
[materialized-views.html%23GUID-8823A06E-853B-4876-AB9F-96D2D4E5A1DE](https://docs.oracle.com/en/database/oracle/oracle-database/19/dwhsg/basic-materialized-views.html%23GUID-8823A06E-853B-4876-AB9F-96D2D4E5A1DE))

نستعرض الـ MV بالشكل التالي:



```

select film_name,customer_name,length_of_renat_in_days from film_rental_view;

```

	FILM_NAME	CUSTOMER_NAME	LENGTH_OF_RENAT_IN_DAYS
1	Movie 10	First Name 32 Last Name 32	32
2	Movie 16	First Name 85 Last Name 85	21
3	Movie 9	First Name 73 Last Name 73	4
4	Movie 3	First Name 82 Last Name 82	43
5	Movie 7	First Name 38 Last Name 38	24
6	Movie 6	First Name 80 Last Name 80	38
7	Movie 11	First Name 22 Last Name 22	14
8	Movie 7	First Name 64 Last Name 64	30
9	Movie 17	First Name 38 Last Name 38	38
10	Movie 5	First Name 91 Last Name 91	45
11	Movie 10	First Name 69 Last Name 69	22
12	Movie 5	First Name 43 Last Name 43	12
13	Movie 9	First Name 77 Last Name 77	41

الطلب الرابع: عرض قائمة بقيم حجوزات الأفلام على مستوى الشهر والبلد والمدينة بكافة الاحتمالات الممكنة.

نقوم بكتابة الاستعلام بالشكل التالي وعرض النتيجة:

```
SELECT
  decode(GROUPING(f.title), 1, 'Multi-Films SUM', f.title) AS film_title,
  decode(GROUPING(t.month), 1, 'Multi-Months SUM', t.month) AS month,
  decode(GROUPING(c.country_name), 1, 'Multi-Countries SUM', c.country_name) AS country,
  decode(GROUPING(c.city_name), 1, 'Multi-Cities SUM', c.city_name) AS city,
  to_char(SUM(r.amount), '9,999,999,999') AS total_amount
FROM
  rental_fact r, film_dim f, customer_dim c, time_dim t
WHERE
  r.customer_id = c.s_customer_key AND r.film_id = f.film_id AND r.rental_date = t.time_id
GROUP BY CUBE(f.title, t.month, c.country_name, c.city_name)
ORDER BY t.month, c.country_name, c.city_name, f.title;
```

Query Result x

SQL | Fetched 50 rows in 0.07 seconds

	FILM_TITLE	MONTH	COUNTRY	CITY	TOTAL_AMOUNT
1	Movie 11	April	Country 1	City 4	1,469
2	Movie 12	April	Country 1	City 4	4,225
3	Multi-Films SUM	April	Country 1	City 4	5,694
4	Movie 18	April	Country 1	City 54	1,006
5	Movie 4	April	Country 1	City 54	707
6	Movie 7	April	Country 1	City 54	2,551
7	Movie 9	April	Country 1	City 54	4,069
8	Multi-Films SUM	April	Country 1	City 54	8,333
9	Movie 11	April	Country 1	Multi-Cities SUM	1,469
10	Movie 12	April	Country 1	Multi-Cities SUM	4,225
11	Movie 18	April	Country 1	Multi-Cities SUM	1,006
12	Movie 4	April	Country 1	Multi-Cities SUM	707



## SELECT

```
to_char(r.RENTAL_DATE,'yyyy') AS Year, f.title AS film,  
SUM(r.amount) OVER (PARTITION BY to_char(r.RENTAL_DATE,'yyyy')) AS Year_Total,  
SUM(r.amount) OVER (PARTITION BY to_char(r.RENTAL_DATE,'yyyy'),  
to_char(r.RENTAL_DATE,'mm')) AS Month_Total,  
DENSE_RANK() OVER (PARTITION BY to_char(r.RENTAL_DATE,'yyyy')  
order by r.amount) AS Year_Rank,  
DENSE_RANK() OVER (PARTITION BY to_char(r.RENTAL_DATE,'yyyy'),  
to_char(r.RENTAL_DATE,'mm') order by r.amount) AS Month_Rank  
FROM rental_fact r, film_dim f WHERE r.film_id = f.film_id;
```

Script Output x Query Result x

SQL | Fetched 150 rows in 0.022 seconds

	YEAR	FILM	YEAR_TOTAL	MONTH_TOTAL	YEAR_RANK	MONTH_RANK
10	2020	Movie 10	3421225	278598	8	2
11	2020	Movie 1	3421225	300503	9	1
12	2020	Movie 12	3421225	352504	10	1
13	2020	Movie 9	3421225	285304	11	1
14	2020	Movie 7	3421225	278598	12	3
15	2020	Movie 8	3421225	284834	13	2
16	2020	Movie 18	3421225	246006	14	1
17	2020	Movie 19	3421225	278598	15	4
18	2020	Movie 4	3421225	352504	16	2
19	2020	Movie 9	3421225	285304	17	2
20	2020	Movie 5	3421225	352504	18	3
21	2020	Movie 11	3421225	270520	19	2
22	2020	Movie 15	3421225	278598	20	5
23	2020	Movie 17	3421225	270520	21	3
24	2020	Movie 4	3421225	278598	22	6

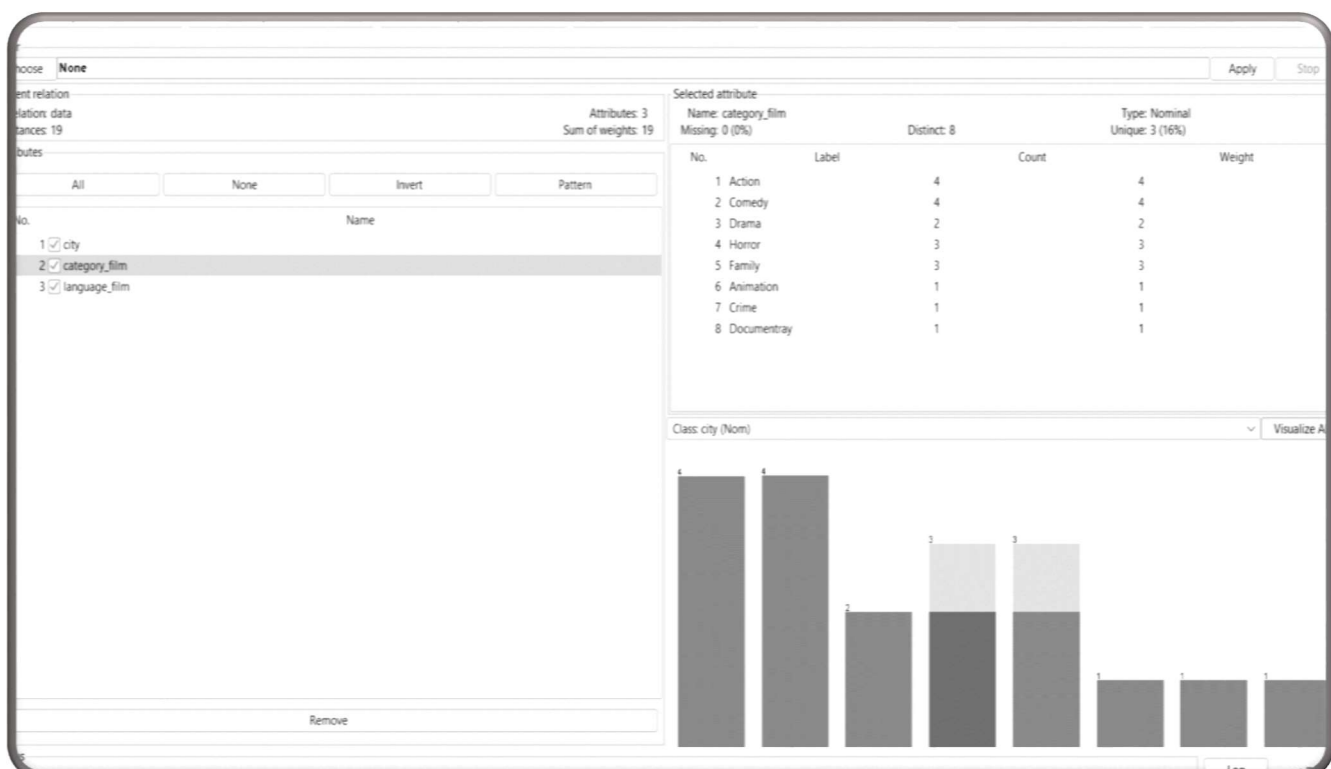
الطلب السادس: يريد مدير الشركة أن يعرف ما هي العالقة ما بين مدينة الزبون وفئة الأفلام ولغة الفيلم لكي يستخدم هذه المعلومة في الإعلانات الموجهة، اقترح طريقة مناسبة لمعرفة هذه العالقة ثم نفذها باستخدام weka.

نملئ البيانات في ملف Excel ونحفظه بصيغة CSV ثم نقوم بفتحه باستخدام weka ونقوم بتحويل الملف من صيغة ARFF الى صيغة CSV

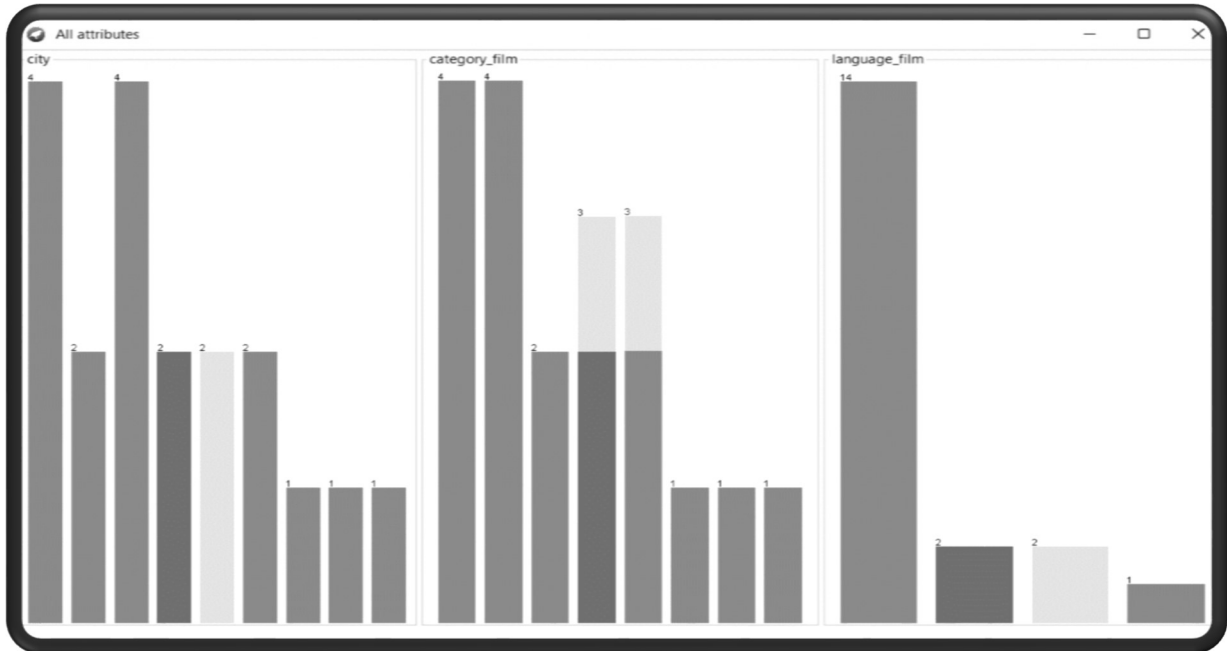
C	B	A
language_film	category_film	city
English	Action	Cario
English	Comedy	New York City
English	Drama	London
French	Horror	Paris
Italian	Family	Rome
English	Comedy	Sydney
English	Action	Cario
Italian	Horror	Rome
Russian	Animation	Moscow
English	Crime	New York City
English	Comedy	London
English	Action	Dubai
English	Action	Cario
English	Drama	London
English	Documentray	Los Angeles
French	Horror	Paris
English	Family	Cario
English	Family	London
English	Comedy	Sydney

البيانات ضمن ال Excel:

فيكون الخرج كالتالي:



وبتحليل جميع البيانات ينتج:



وبتنفيذ خوارزمية apriori على البيانات المدخلة تكون:

```

Associator output

Scheme:      weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation:    data
Instances:    19
Attributes:   3
              city
              category_film
              language_film
=== Associator model (full training set) ===

Apriori
=====

Minimum support: 0.1 (2 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 18

Generated sets of large itemsets:

Size of set of large itemsets L(1): 14
Size of set of large itemsets L(2): 15
Size of set of large itemsets L(3): 4

Best rules found:

1. city=Carrio 4 ==> language_film=English 4    <conf:(1)> lift:(1.36) lev:(0.06) [1] conv:(1.05)
2. city=London 4 ==> language_film=English 4    <conf:(1)> lift:(1.36) lev:(0.06) [1] conv:(1.05)
3. category_film=Action 4 ==> language_film=English 4    <conf:(1)> lift:(1.36) lev:(0.06) [1] conv:(1.05)
4. category_film=Comedy 4 ==> language_film=English 4    <conf:(1)> lift:(1.36) lev:(0.06) [1] conv:(1.05)
5. city=Carrio category_film=Action 3 ==> language_film=English 3    <conf:(1)> lift:(1.36) lev:(0.04) [0] conv:(0.79)
6. city=New York City 2 ==> language_film=English 2    <conf:(1)> lift:(1.36) lev:(0.03) [0] conv:(0.53)
7. category_film=Drama 2 ==> city=London 2    <conf:(1)> lift:(4.75) lev:(0.08) [1] conv:(1.58)
8. city=Paris 2 ==> category_film=Horror 2    <conf:(1)> lift:(6.33) lev:(0.09) [1] conv:(1.68)
9. language_film=French 2 ==> city=Paris 2    <conf:(1)> lift:(9.5) lev:(0.09) [1] conv:(1.79)
10. city=Paris 2 ==> language_film=French 2    <conf:(1)> lift:(9.5) lev:(0.09) [1] conv:(1.79)
  
```