Solutions

Laboratory 2b – Star Schemas

PART I: The USELOG Case Study

Description

University Computer Lab's director keeps track of the lab usage, measured by the number of students using the lab. This particular function is very important for budgeting purposes. The computer lab director assigns you the task of developing a small Data Warehouse in which to keep track of the lab usage statistics. The main requirements for this database are to:

a. Show the usage numbers by different time periods (e.g. morning, afternoon, night)

b. Show the usage numbers by time period (e.g. morning, afternoon, night), by major, and by student's class

c. Compare the usage numbers for different majors and semesters (e.g. semester 1, semester 2).

Use the provided database that includes the following tables: USELOG, STUDENT, MAJOR, and CLASS

USELOG contains the student access data
USELOG (Log_Date, Log_Time, Student_ID, Act)

STUDENT is a table containing student data
STUDENT (Student_ID, Sex, Full/Part, Type, Class_ID, Major_Code)

MAJOR is a table containing major data
MAJOR (Major_Name, Major_Code)

CLASS is a table containing class data
CLASS (Class_Description, Class_ID)

Tasks

Given the requirements above, complete the following:

1. Create a star schema for the USELOG data.
2. Define the dimensions and attributes for the USELOG star schema.
3. Write the SQL statements for the implementation of the star schema.
   The following operational databases have been provided for you:

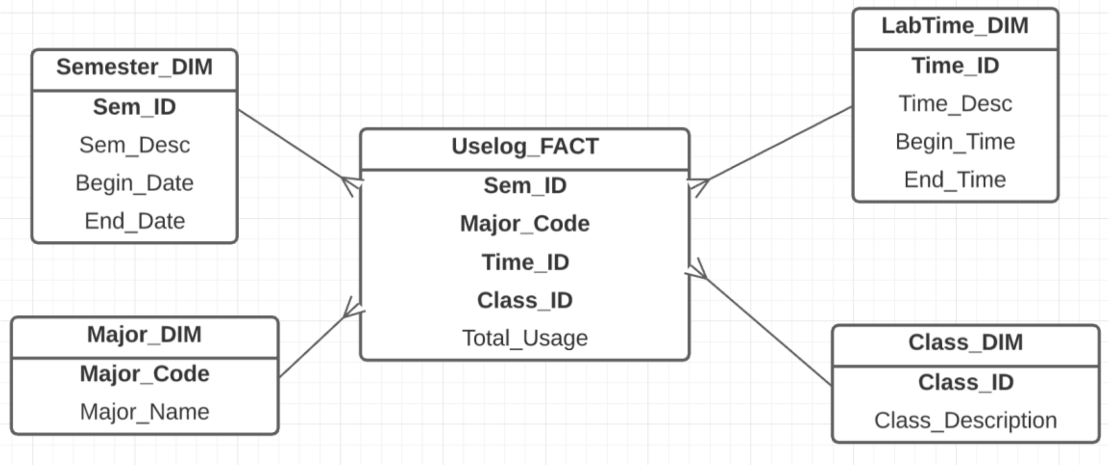
   `dw.Class`: table that stores information about classification ids and descriptions
   `dw.Major`: table that stores information about major codes and descriptions
   `dw.Student`: table that stores information about students as described above
   `dw.Uselog`: table that stores information about lab usage as described above

   You do not need to copy these four tables (**`dw.Class`**, **`dw.Major`**, **`dw.Student`**, and **`dw.Uselog`**) into your account. You can just simply use these tables.

4. Write the SQL statements to produce the following reports:
   a. Show the usage numbers by different time periods (e.g. morning, afternoon, night)
   b. Show the usage numbers by time period (e.g. morning, afternoon, night), by major, and by student's class
   c. Show the usage numbers for different majors and semesters (e.g. semester 1, semester 2).

Solutions

Tasks: Q1 and Q2: Star Schema



Task: Q3 The SQL to Create the Dimensions and Fact

```
------------------------------------------------------
-- Step 1: create the dimensions
------------------------------------------------------
-- create semester dimension
DROP TABLE Semester_DIM CASCADE CONSTRAINTS PURGE;
CREATE TABLE Semester_DIM (
   Sem_ID       VARCHAR2(10),
   Sem_Desc     VARCHAR2(20),
   Begin_Date   DATE,
   End_Date     DATE
);

-- create labtime dimension
DROP TABLE LabTime_DIM CASCADE CONSTRAINTS PURGE;
CREATE TABLE LabTime_DIM (
   Time_ID      NUMBER,
   Time_Desc    VARCHAR2(15),
   Begin_Time   DATE,
   End_Time     DATE
);

-- create major dimension
DROP TABLE Major_DIM CASCADE CONSTRAINTS PURGE;
CREATE TABLE Major_DIM AS
SELECT * FROM dw.Major;

-- create class dimension
DROP TABLE Class_DIM CASCADE CONSTRAINTS PURGE;
CREATE TABLE Class_DIM AS
SELECT * FROM dw.Class;

-- populate semester dimension for Semester 1 and Semester 2
-- (the begin and end date can be changed according to the case)
INSERT INTO Semester_DIM
   VALUES ('S1', 'Semester1', TO_DATE('01-JAN', 'DD-MON'),
         TO_DATE('15-JUL', 'DD-MON'));
INSERT INTO Semester_DIM
   VALUES ('S2', 'Semester2', TO_DATE('16-JUL', 'DD-MON'),
         TO_DATE('31-DEC', 'DD-MON'));

--populate labtime dimension
-- populate labtime dimension for morning, afternoon and night
-- (the begin and end time can be changed according to the case)
INSERT INTO LabTime_DIM
   VALUES(1, 'morning', TO_DATE('06:01', 'HH24:MI'),
         TO_DATE('12:00', 'HH24:MI'));
INSERT INTO LabTime_DIM
   VALUES(2, 'afternoon', TO_DATE('12:01', 'HH24:MI'),
         TO_DATE('18:00', 'HH24:MI'));
```

```
INSERT INTO LabTime_DIM
    VALUES(3, 'night', TO_DATE('18:01', 'HH24:MI'),
            TO_DATE('06:00', 'HH24:MI'));

----------------------------------------------------
-- Step 2: create a temp fact table to extract from
-- uselog table
----------------------------------------------------
DROP TABLE Temp_Fact_Uselog CASCADE CONSTRAINTS PURGE;
CREATE TABLE Temp_Fact_Uselog AS
    SELECT u.Log_Date , u.Log_Time, u.Student_ID, S.Class_ID, S.Major_Code
    FROM dw.Uselog u, dw.Student S
    WHERE u.Student_ID = S.Student_ID;

-- add a column in the tempfact table to store timeid
-- (cannot directly do this in the tempfact table because
-- log_time was of DATE type and timeid is of NUMBER type).
ALTER TABLE temp_fact_uselog
ADD (Time_ID NUMBER);

UPDATE temp_fact_uselog
SET Time_ID = 1
WHERE  to_char(Log_Time, 'HH24:MI') >= '06:01'
AND to_char(Log_Time, 'HH24:MI') <='12:00';

UPDATE temp_fact_uselog
SET Time_ID = 2
WHERE  to_char(Log_Time, 'HH24:MI') >= '12:01'
AND to_char(Log_Time, 'HH24:MI') <='18:00';

-- note that we use OR in the last update statement to
-- include the time between 18:01 and 06:00.
UPDATE temp_fact_uselog
SET Time_ID = 3
WHERE to_char(Log_Time, 'HH24:MI') >= '18:01'
OR to_char(Log_Time, 'HH24:MI') <='06:00';

-- alternatively, you may want to update timeid=3
-- for all other records where the time_id is still empty
-- update temp_fact_uselog
-- set timeid = 3
-- where timeid is NULL;

-- add a column in the temp_fact_uselog table to store semid
-- (cannot directly do this in the test table because
-- log_date was of DATE type and semid is of VARCHAR type.)
ALTER TABLE temp_fact_uselog
ADD (Sem_ID VARCHAR2(10));
```

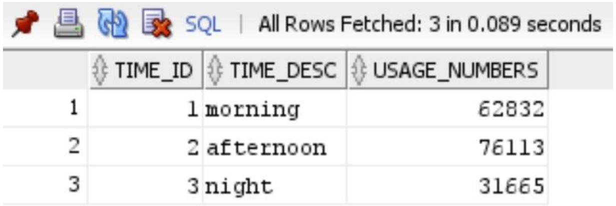
```
-- populate the new attribute semid by summarizing
-- the date(log_date)
UPDATE temp_fact_uselog
SET Sem_ID = 'S1'
WHERE to_char(Log_Date, 'MMDD') >= '0101'
AND to_char(Log_Date, 'MMDD') <= '0715';

UPDATE temp_fact_uselog
SET Sem_ID = 'S2'
WHERE to_char(Log_Date, 'MMDD') >= '0716'
AND to_char(Log_Date, 'MMDD') <= '1231';


-- Now, create the fact table,
-- make sure to include the TOTAL aggregate.
-- This is an aggregate table of the earlier tempfact table.
DROP TABLE Uselog_FACT CASCADE CONSTRAINTS PURGE;
CREATE TABLE Uselog_FACT AS
SELECT T.Sem_ID, T.Time_ID, T.Class_ID,
      T.Major_Code, COUNT(T.Student_ID) AS Total_Usage
FROM temp_fact_uselog T
GROUP BY T.Sem_ID, T.Time_ID, T.Class_ID, T.Major_Code;
```

Note: In this solution, we are using the Temporary Fact table as we have two manually created dimensions: Semester Dimension and Lab Time Dimension.

Task: Q4 The Reports

a) Show the usage numbers by different time periods (e.g. morning, afternoon, night)

```
SELECT u.Time_ID, L.Time_Desc, SUM(u.Total_Usage) AS Usage_Numbers
FROM Uselog_FACT u, LabTime_DIM L
WHERE u.Time_ID = L.Time_ID
GROUP BY u.Time_ID, L.Time_Desc
ORDER BY u.Time_ID;
```



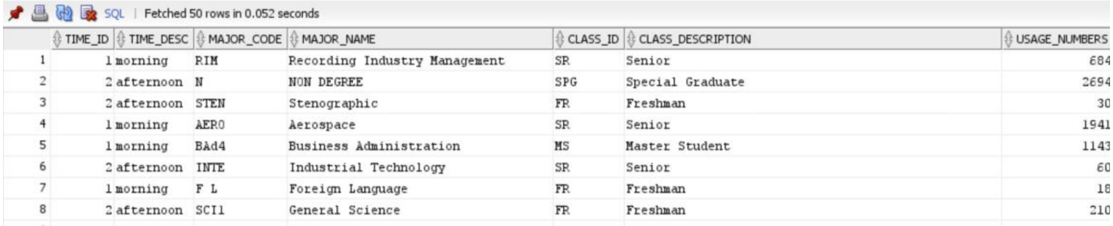
All Rows Fetched: 3 in 0.089 seconds

	TIME_ID	TIME_DESC	USAGE_NUMBERS
1	1	morning	62832
2	2	afternoon	76113
3	3	night	31665

```
3 rows selected.
```

b) Show the usage numbers by time period (e.g. morning, afternoon, night), by major, and by student's class)

```
SELECT u.Time_ID, L.Time_Desc, u.Major_Code, M.Major_Name, u.Class_ID,
    C.Class_Description, SUM(u.Total_Usage) AS Usage_Numbers
FROM Uselog_FACT u, LabTime_DIM L, Major_DIM M, Class_DIM C
WHERE u.Time_ID = L.Time_ID
AND u.Major_Code = M.Major_Code
AND u.Class_ID = C.Class_ID
GROUP BY u.Time_ID, L.Time_Desc, u.Major_Code, M.Major_Name, u.Class_ID,
    C.Class_Description;
```

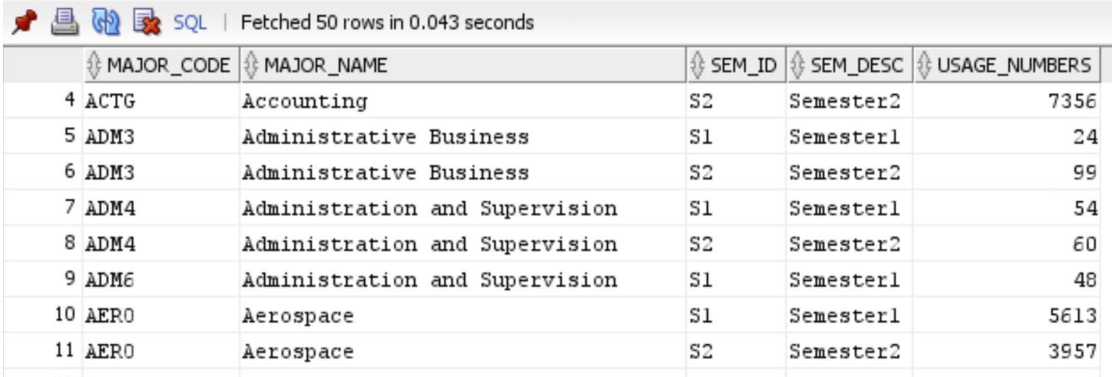


	TIME_ID	TIME_DESC	MAJOR_CODE	MAJOR_NAME	CLASS_ID	CLASS_DESCRIPTION	USAGE_NUMBERS
1	1	morning	RIM	Recording Industry Management	SR	Senior	684
2	2	afternoon	N	NON DEGREE	SPG	Special Graduate	2694
3	2	afternoon	STEN	Stenographic	FR	Freshman	30
4	1	morning	AERO	Aerospace	SR	Senior	1941
5	1	morning	BAd4	Business Administration	MS	Master Student	1143
6	2	afternoon	INTE	Industrial Technology	SR	Senior	60
7	1	morning	F L	Foreign Language	FR	Freshman	18
8	2	afternoon	SCI1	General Science	FR	Freshman	210

```
773 rows selected.
```

c) Show the usage numbers for different majors and semesters (e.g. semester 1, semester 2).

```
SELECT u.Major_Code, M.Major_Name, u.Sem_ID, S.Sem_Desc,
    SUM(u.Total_Usage) AS Usage_Numbers
FROM Uselog_FACT u, Major_DIM M, Semester_DIM S
WHERE u.Major_Code = M.Major_Code
AND u.Sem_ID = S.Sem_ID
GROUP BY u.Major_Code, M.Major_Name, u.Sem_ID, S.Sem_Desc
ORDER BY u.Major_Code;
```



	MAJOR_CODE	MAJOR_NAME	SEM_ID	SEM_DESC	USAGE_NUMBERS
4	ACTG	Accounting	S2	Semester2	7356
5	ADM3	Administrative Business	S1	Semester1	24
6	ADM3	Administrative Business	S2	Semester2	99
7	ADM4	Administration and Supervision	S1	Semester1	54
8	ADM4	Administration and Supervision	S2	Semester2	60
9	ADM6	Administration and Supervision	S1	Semester1	48
10	AERO	Aerospace	S1	Semester1	5613
11	AERO	Aerospace	S2	Semester2	3957

```
207 rows selected.
```

PART II: The ROBCOR Aviation Charters Case Study

Description

ROBCOR, Inc. provides “on demand” aviation charters, using a mix of different airplane and airplane types. Because ROBCOR, Inc., has grown rapidly, it has hired you to be its first database manager. Your first and critical assignment is to develop a decision support system to analyze the charter data.

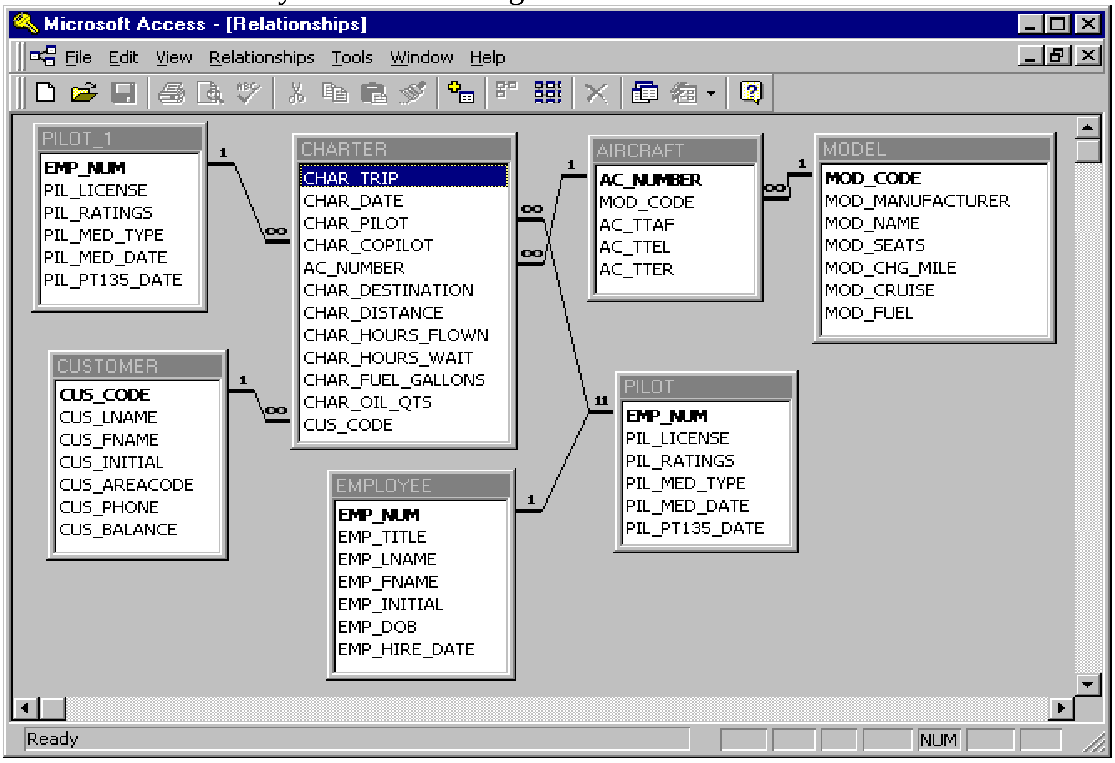
The charter operations manager wants to be able to analyze charter data such as total hours flown, total fuel used, and total revenue (charter distance x model charge per mile). She would also like to be able to drill-down by pilot, airplane model, and time periods. The main requirements for this database are to:

a. Show the total revenue each month/year

b. Show the total hours flown by each pilot

c. Show the total fuel used by each airplane model.

The database currently has the following tables:



PILOT_1
EMP_NUM
PIL_LICENSE
PIL_RATINGS
PIL_MED_TYPE
PIL_MED_DATE
PIL_PT135_DATE

CHARTER
CHAR_TRIP
CHAR_DATE
CHAR_PILOT
CHAR_COPILOT
AC_NUMBER
CHAR_DESTINATION
CHAR_DISTANCE
CHAR_HOURS_FLOWN
CHAR_HOURS_WAIT
CHAR_FUEL_GALLONS
CHAR_OIL_QTS
CUS_CODE

AIRCRAFT
AC_NUMBER
MOD_CODE
AC_TTAF
AC_TTEL
AC_TTER

MODEL
MOD_CODE
MOD_MANUFACTURER
MOD_NAME
MOD_SEATS
MOD_CHG_MILE
MOD_CRUISE
MOD_FUEL

CUSTOMER
CUS_CODE
CUS_LNAME
CUS_FNAME
CUS_INITIAL
CUS_AREACODE
CUS_PHONE
CUS_BALANCE

PILOT
EMP_NUM
PIL_LICENSE
PIL_RATINGS
PIL_MED_TYPE
PIL_MED_DATE
PIL_PT135_DATE

EMPLOYEE
EMP_NUM
EMP_TITLE
EMP_LNAME
EMP_FNAME
EMP_INITIAL
EMP_DOB
EMP_HIRE_DATE

The tables can be copied from the dw account, using:

```
      Select * from dw.<table_name>;
```

Or

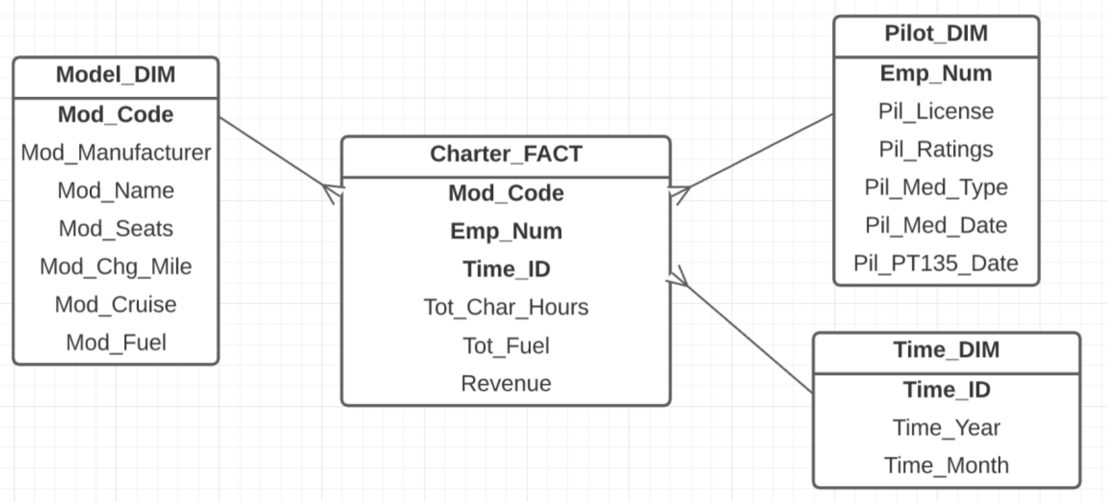
```
      Create Table <your_table_name> As
      Select …
      From dw.<table_name>
      Where …
```

Given these requirements, complete the following:

1. Create a star schema for the charter data.
2. Define the dimensions and attributes for the charter operation's star schema.
3. Define the SQL statements for the implementation of the star schema.
4. Write the SQL statements to produce the following reports:
    a. Show the total revenue each year
    b. Show the total hours flown by each pilot
    c. Show the total fuel used by each aircraft model

Solutions

Tasks: Q1 and Q2: Star Schema



Task: Q3 The SQL to Create the Dimensions and Fact

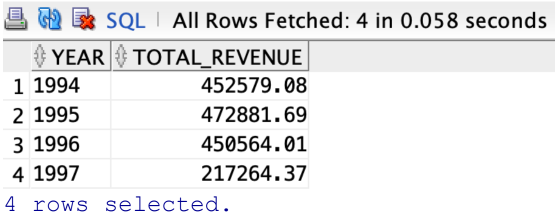
```
--------------------------------------------------------
-- Step 1: create the dimensions
--------------------------------------------------------
--create time dimension
DROP TABLE Time_DIM CASCADE CONSTRAINTS PURGE;
CREATE TABLE Time_DIM AS
SELECT DISTINCT to_char(Char_Date, 'YYYYMM') AS Time_ID,
            to_char(Char_Date, 'Month') AS Time_Month,
            to_char(Char_Date, 'YYYY') AS Time_Year
FROM dw.Charter;

--copy model and pilot dimensions
DROP TABLE Model_DIM CASCADE CONSTRAINTS PURGE;
CREATE TABLE Model_DIM AS
SELECT * FROM dw.Model;

DROP TABLE Pilot_DIM CASCADE CONSTRAINTS PURGE;
CREATE TABLE Pilot_DIM AS
SELECT * FROM dw.Pilot;

--------------------------------------------------------
```

```
-- Step 2: create the Charter_fact (the fact table)
-----------------------------------------------------
CREATE TABLE Charter_FACT AS
SELECT C.Char_Pilot AS Emp_Num,
       M.Mod_Code,
       to_char(C.Char_Date, 'YYYYMM') AS Time_ID,
       SUM(C.Char_Hours_Flown) AS Tot_Char_Hours,
       SUM(C.Char_Fuel_Gallons) AS Tot_Fuel,
       SUM(C.Char_Distance * M.Mod_Chg_Mile) AS Revenue
FROM   dw.Charter C, dw.Model M, dw.Aircraft A
WHERE  C.Ac_Number=A.Ac_Number AND A.Mod_Code=M.Mod_Code
GROUP BY C.Char_Pilot, M.Mod_Code, to_char(C.Char_Date, 'YYYYMM');
```

Task: Q4 The Reports

a. Show the total revenue each year

```
SELECT T.Time_Year AS Year, SUM(C.Revenue) AS Total_Revenue
FROM Charter_FACT C, Time_DIM T
WHERE C.Time_ID = T.Time_ID
GROUP BY T.Time_Year
ORDER BY T.Time_Year;
```



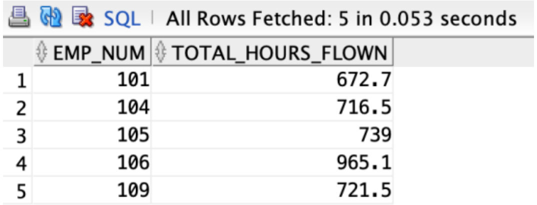
All Rows Fetched: 4 in 0.058 seconds

	YEAR	TOTAL_REVENUE
1	1994	452579.08
2	1995	472881.69
3	1996	450564.01
4	1997	217264.37

```
4 rows selected.
```

b. Show the total hours flown by each pilot

```
SELECT Emp_Num, SUM(Tot_Char_Hours) AS Total_Hours_Flown
FROM  Charter_Fact
GROUP BY Emp_Num
ORDER BY Emp_Num;
```



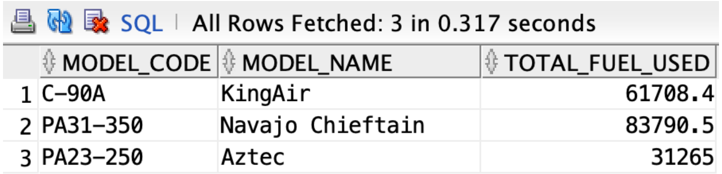
SQL | All Rows Fetched: 5 in 0.053 seconds

	EMP_NUM	TOTAL_HOURS_FLOWN
1	101	672.7
2	104	716.5
3	105	739
4	106	965.1
5	109	721.5

```
5 rows selected.
```

c. Show the total fuel used by each aircraft model.

```
SELECT M.Mod_Code AS Model_Code, M.Mod_Name AS Model_Name,
    SUM(C.Tot_Fuel) AS Total_Fuel_Used
FROM  Charter_Fact C, Model_DIM M
WHERE M.Mod_Code = C.Mod_Code
GROUP BY M.Mod_Code, M.Mod_Name;
```



SQL | All Rows Fetched: 3 in 0.317 seconds

	MODEL_CODE	MODEL_NAME	TOTAL_FUEL_USED
1	C-90A	KingAir	61708.4
2	PA31-350	Navajo Chieftain	83790.5
3	PA23-250	Aztec	31265

```
3 rows selected.
```