Analytics refers to the field of data analysis. It involves discovery of meaningful information (in terms of correlations, trends, patterns, etc.) in data via data queries as well as systematic computational and/or statistical analysis of data. It often involves studying past historical data to discover potential trends, to analyze the effects of certain decisions or events, or to evaluate the performance of a given business scenario. Data analytics (DA) is the science or process of examining raw data to uncover hidden patterns, unknown correlations and other useful information that can be used to make better decisions and planning. Spectrum of Data Analytics Descriptive analytics: It encompasses the set of techniques that summarizes and describes what has happened in the past. (WHAT HAPPENED) Tools/MethodsData queries & reporting Descriptive statistics Data visualization (data dashboards, etc.) What-if spreadsheet models Predictive analytics: It consists of techniques that use models constructed from past data to predict the future or ascertain the impact of one variable on another. (WHAT MIGHT HAPPEN?) Ex: Analytics of survey data of past purchase behavior can be used to help predict the market share of a new product. Prescriptive Analytics: It indicates a best course of action to take (WHAT TO DO to make it happen and prevent it from happening?) Referred to as the "final frontier of analytic capabilities,“prescriptive analytics automatically synthesizes big data, multiple disciplines of mathematical sciences and computational sciences, and business rules, to make predictions and then suggests decision options and actions to take advantage of the predictions. Single-row operators used with single-row subqueries include: =, >, <, >=, <=, and <>. Multi-row operators that can be used with multiple-row subqueries include IN, ALL, What is true about multi-row subqueries? .They return more than one row of data from executing the subquery. Which of the following is true regarding inserting a subquery in an outer query? All of the above. A subquery must be placed in the outer query's HAVING clause if: The value returned by the inner query is to be compared to grouped data in the outer query. A subquery must be placed in the outer query's FROM clause if: The inner query is used to create a temporary table.. A subquery cannot contain a(n) ORDER BY clause except that it is nested in a FROM clause. Which of the following situations that a subquery is suitable? When you need to find the titles of all books shipped on the same date as an order placed by a particular customer. Which of the following statements is true about subqueries? Subqueries are executed before the outer query is executed. Which of the following is true about the result of a subquery? The result of a subquery is used by the outer query. Which of the following clause is mandatorily used in a subquery? (multiple answers) SELECTFROM In which of the following clauses can a subquery be used? All of the above We are to find the salaries for all the employees who have a higher salary than the IT manager. Which of the following queries will produce the required result? SELECT firstname, lastname, monthly\_salary FROM emp WHERE monthly\_salary > (SELECT monthly\_salary FROM emp WHERE jobtitle = ‘IT MANAGER’); Find the employees whose salary is equal to the salary of at least one employee in Department 10. (i.e., equal to any of the list) =ANY or IN Find out all the employees who have salary greater than at least one employee in Department 10. (i.e., higher than the lowest) >ALL Find out all the employees who have salary lesser than the salary of all the employees in Department 10. (i.e., lower than the lowest) <ANY The following two queries will prompt an error message. What’s wrong with them? SELECT empno, monthly\_salary FROM emp WHERE monthly\_salary = (SELECT monthly\_salary FROM emp WHERE deptno = 10); The equals sign produces too many records so it needs to be changed to =ANY SELECT lastname FROM emp WHERE jobtitle = (SELECT jobtitle FROM emp WHERE empno < 7500); Same error as previous question What will be the outcome of the following query? SELECT lastname, jobtitle, monthly\_salary FROM emp WHERE monthly\_salary = (SELECT monthly\_salary FROM emp WHERE jobtitle = 'CLERK') AND jobtitle <> 'CLERK'; It will not execute successfully and prompt an error message “single-row subquery returns more than one row.” What will be the outcome of the following query? SELECT lastname, jobtitle, monthly\_salary FROM emp WHERE monthly\_salary >= (SELECT monthly\_salary FROM emp WHERE jobtitle = 'CLERK') AND jobtitle <> 'CLERK'; It will not execute successfully and prompt an error message “single-row subquery returns more than one row.” What will be the outcome of the following query? SELECT lastname, jobtitle, monthly\_salary FROM emp WHERE monthly\_salary >ALL (SELECT monthly\_salary FROM emp WHERE jobtitle = 'CLERK') AND jobtitle <> 'CLERK' ORDER BY monthly\_salary; It returns those non-clerk employees whose monthly pay is higher than the highest-paid clerk. What will be the outcome of the following query? SELECT lastname, jobtitle, monthly\_salary FROM emp WHERE monthly\_salary <ANY (SELECT monthly\_salary FROM emp WHERE jobtitle = 'CLERK') AND jobtitle <> 'CLERK' ORDER BY monthly\_salary; It returns those non-clerk employees whose monthly pay is less than the highest-paid clerk. What will be the outcome of the following query? SELECT firstname, lastname, monthly\_salary FROM emp WHERE monthly\_salary > (SELECT MAX(monthly\_salary) FROM emp WHERE deptno = 30); It executes successfully and gives the names and salary for those employees whose salary is higher than the highest paid employee in Department 30. The following query will prompt an error. Choose the correct reason for the error as given in the options. (multiple answers) SELECT firstname, lastname FROM emp WHERE commission IN (SELECT MIN(commission) FROM emp GROUP BY deptno); The single row subquery gives multiple records The use of "=" operator is invalid; an IN operator will work correctly What will be the outcome of the following query? SELECT empno, firstname, lastname, jobtitle, monthly\_salary FROM emp WHERE monthly\_salary < ALL (SELECT monthly\_salary FROM emp WHERE jobtitle = 'SALESMAN') AND jobtitle <> 'SALESMAN'; Return the employees who are not a sales person and have a salary less than that of the salary for all of the salespersons. The subquery inserted in the query above is a multiple-row subquery. Which of the following queries is valid? SELECT firstname, lastname, monthly\_salary, deptno FROM emp WHERE monthly\_salary = (SELECT MIN(monthly\_salary) FROM emp); SELECT firstname, lastname, monthly\_salary, deptno FROM emp WHERE monthly\_salary =AN Is the following 2 queries equivalent? SELECT firstname, lastname, monthly\_salary, deptno FROM emp WHERE monthly\_salary = (SELECT MIN(monthly\_salary) FROM emp); SELECT firstname, lastname, monthly\_salary, deptno FROM emp WHERE monthly\_salary = MIN(monthly\_salary); No What will be the outcome of the following query? SELECT lastname, jobtitle, monthly\_salary FROM emp WHERE monthly\_salary >ALL (SELECT monthly\_salary FROM emp WHERE jobtitle = 'SALESMAN') ORDER BY monthly\_salary; The people who are not a salesperson and has a salary lower than all of the sales people. SELECT lastname, jobtitle, monthly\_salary FROM emp WHERE monthly\_salary =ANY (SELECT monthly\_salary FROM emp WHERE jobtitle = 'SALESMAN') A nonsalesperson who has a salary equal to any salespersons salary What will be the outcome of the following query? SELECT category, title, cost FROM books WHERE cost < (SELECT cost FROM books WHERE title = 'DATABASE IMPLEMENTATION') AND category = 'COMPUTER' ORDER BY cost; It selects the book that costs less than DB implementation and gives the category, title and cost of that book What will be the outcome of the following query? SELECT title, retail FROM books WHERE cost = (SELECT MAX(cost) FROM books); It finds the book that costs the most and outputs the name of that book along with the price of the book We are to find out the most expensive book. Which of the following queries will give you the required results? SELECT title, retail FROM books WHERE retail = (SELECT MAX(retail) FROM books); Which of the following is syntactically correct that involves a subquery? SELECT order# FROM orders WHERE shipdate = (SELECT shipdate FROM orders WHERE order# = 1010); Which of the following statements is valid? SELECT title FROM books WHERE cost < (SELECT cost FROM books WHERE isbn = '9959789321'); Which of the following queries is a valid query to identify those customers who live in the same state as the customer named Leila Smith? SELECT customer# FROM customers WHERE state = (SELECT state FROM customers WHERE lastname = 'SMITH' AND firstname = 'LEILA'); Study the following SQL statement: SELECT customer# FROM customers WHERE referred = (SELECT referred FROM customers WHERE firstname = 'JORGE' AND lastname = 'PEREZ'); What is the purpose of the SQL statement listed above? To identify other customers who were referred to JustLee Books by the same person who referred Jorge Perez. We are to find out those books with a retail price greater than the average retail price of other books in the same category.SELECT b.title, b.retail, t.category, t.cataverage FROM books b, (SELECT category, AVG(retail) cataverage FROM books GROUP BY category) t WHERE b.category = t.category; Exam the following query: SELECT b.title, b.cost, t.category, t.catalog\_max FROM books b, (SELECT category, MAX(cost) catalog\_max FROM books GROUP BY category) t WHERE b.category = t.category; The inner query above is considered a multiple-column subquery.Determine which orders were shipped on the same state as order 1014. (Also include the ship state and ship date in the query result.) SELECT Order#, Shipstate, Shipdate FROM orders WHERE Shipstate = (SELECT shipstate FROM orders WHERE order# ='1014'); Find out the most expensive book sold by JustLee. (Show the title of the book, along with its retail price.) SELECT title,retail FROM books WHERE retail =(SELECT MAX(retail) FROM books); Find out the book that was most recently published. (List the book title and publication date.) SELECT title,pubdate FROM books WHERE pubdate =(SELECT MAX(pubdate) FROM books); List the book title and retail price for all books with a retail price higher than the average retail price of all books sold by JustLee Books. Also include the average retail price of all books in the query result and sort the data by the retail price. SELECT title,retail,(SELECT round(AVG(retail), 2) FROM books) "AVG\_Retail" FROM books WHERE retail>(SELECT round(AVG(retail), 2) FROM books) GROUP BY title, retail; List the title of those books published by the publisher of the book “Big Bear and Little Dove” and those books that generate more than the average profit returned by all books sold by JustLee Books. SELECT isbn, title FROM books WHERE pubid = (SELECT pubid FROM books WHERE title = ‘BIG BEAR AND LITTLE DOVE’) and (retail-cost))(SELECT avg(retail-cost) FROM books); Find out those computer books that have a higher average profit than that of the book 'E-BUSINESS THE EASY WAY' (Note: profit = retail – cost). SELECT title, avg(retail-cost) AS “AVG\_PROFIT” FROM books WHERE category =”COMPUTER” GROUP BY title HAVING avg(retail-cost))(SELECT avg(retail-cost) FROM books WHERE title = “E-NUSINESS THE EASY WAY”); Problem 1. (Debugging) Examine the following two queries and pinpoint the errors. How do you fix them? SELECT category, title, cost FROM books WHERE cost >= SELECT cost FROM books WHERE title = 'DATABASE IMPLEMENTATION' ORDER BY cost; --Added Parentheses around select statement SELECT category, title, cost FROM books WHERE (SELECT cost FROM books WHERE title = 'DATABASE IMPLEMENTATION') <= cost ORDER BY cost; --Put cost statement in the front, otherwise functions correctly. Problem 2. Create a query to find out those computer books that were published after the book titled “HANDCRANKED COMPUTERS” was published. Sort the records in ascending order of pubdate. SELECT category, title, pubdate FROM books WHERE pubdate > (select p ORDER BY pubdate; SELECT category, title, pubdate FROM books WHERE category = 'COMPUTER' and pubdate > (select pubdate from books where title = 'HANDCRANKED COMPUTERS') ORDER BY pubdate asc; SELECT cost FROM books WHERE title ='DATABASE IMPLEMENTATION'; SELECT category, title, cost FROM books WHERE cost >= (SELECT cost FROM books WHERE title ='DATABASE IMPLEMENTATION') ORDER BY cost; --This code does not work will will discuss at a later time SELECT title, shipdate, customer# FROM customers JOIN orders USING(customer#) JOIN orderitems USING(order#) JOIN books USING(isbn) WHERE shipdate <= (SELECT shipdate FROM orders WHERE customer# = 'C1005'); SELECT category, title, pubdate FROM books WHERE pubdate >(SELECT pubdate FROM books WHERE title = 'HANDCRANKED COMPUTERS') AND category='COMPUTER' ORDER BY pubdate; SELECT category, title, cost FROM books WHERE (SELECT cost FROM books WHERE title = 'DATABASE IMPLEMENTATION') <= cost ORDER BY cost; SELECT category, title, cost FROM books WHERE cost >=( SELECT cost FROM books WHERE title = 'DATABASE IMPLEMENTATION') ORDER BY cost; SELECT category, title, cost FROM books WHERE cost >=(SELECT cost FROM books WHERE title='DATABASE IMPLEMENTATION') AND category ='COMPUTER' ORDER BY cost; --INCORRECT SELECT title,MIN(retail) FROM books; --INCORRECT SELECT title,MIN(retail) FROM books GROUP BY title; --INCORRECT SELECT MIN(retail) FROM books; --CORRECT SELECT title,retail FROM books WHERE retail =(SELECT MIN(retail) FROM books); SELECT category, AVG(retail-cost) "Average Profit" FROM books GROUP BY category HAVING AVG(retail-cost) > (SELECT AVG(retail-cost) FROM books WHERE category='LITERATURE'); --SUBQUERY SELECT ROUND(AVG(retail), 2) FROM books; SELECT title, retail, (SELECT ROUND(AVG(retail), 2) FROM books) "Average Retail" FROM books; SELECT category, ROUND(AVG(retail),2) cataverage FROM books GROUP BY category SELECT b.title, b.reatil, t.cataveragefrom books b, SELECT category, ROUND(AVG(retail),2) cataverage FROM books GROUP BY category Where b.category = t.category The output can be used as a temporary table that stores data of the average retail price for each category. Write a query as an alternative solution to Problem 7 (in Lecture 3) with the JOIN … USING method. SELECT DISTINCT b.title, b.retail, category, t.cataverage FROM books b JOIN (select category, ROUND(AVG(retail),2) cataverage from books group by category) t USING (category) Where b.retail > t.cataverage group by b.title, b.retail, category, t.cataverage; Does the following query work? If not, why? How do you fix the errors? SELECT lastname, jobtitle, monthly\_salary FROM emp WHERE monthly\_salary >ALL (SELECT monthly\_salary FROM emp WHERE jobtitle = 'CLERK') AND jobtitle <> 'CLERK' ORDER BY monthly\_salary; The single row function in the subquery produces a result of more than one row. To fix this a >ALL function is needed Write a query as an alternative solution to Problem 9 (without using <ALL) SELECT lastname, jobtitle, monthly\_salary FROM emp WHERE monthly\_salary < (SELECT max(monthly\_salary) FROM emp WHERE jobtitle = 'CLERK') AND jobtitle <> 'CLERK' ORDER BY monthly\_salary; Write a query as an alternative solution to Problem 11 (without using <ANY) select lastname, jobtitle, monthly\_salary from emp where monthly\_salary < (select MAX(monthly\_salary) FROM emp WHERE jobtitle = 'CLERK') and jobtitle <> 'CLERK' order by monthly\_salary; Is the following query valid? If not, why? How do you fix the errors? SELECT title, retail, category FROM books WHERE retail = (SELECT MAX(retail) FROM books) ORDER BY category; There is a group by function in the subquery that should not be there to fix this issue take out the GROUP BY. Is the following query valid? If not, why? How do you fix the errors? SELECT title, retail, category FROM books WHERE retail IN (SELECT category, MAX(retail) FROM books GROUP BY category) ORDER BY category; The query is trying to produce too many values to fix this the category needs to be removed from the subquery Find out those books (in any category) that have a retail price less than the highest retail value of the cooking books. SELECT b1.title, b1.category, b1.retail, (SELECT max(retail) FROM books where category = ‘COOKING’) AS ‘HIGHEST RETAIL COOKING’ FROM books b1, books b2 WHERE b1.isbn = b2.isbn and b1.retail <ANY (SELECT retail FROM books WHERE category = ‘COOKING’) ORDER BY category; Find out those computer books with a retail price higher than the highest retail value of the FITNESS books. SELECT b1.title, b1.category, b1.retail, (SELECT MAX(retail) FROM books WHERE category = ‘FITNESS’) AS “HIGHEST RETAIL-FITNESS” FROM books b1, books b2 WHERE b1.isbn = b2.isbn and b1.retail >ALL (SELECT retail FROM books WHERE category = ‘FITNESS’) AND b1.category = ‘COMPUTER’ ORDER BY category; Searches for the books purchased by the customers who also purchased the book with ISBN 0401140733. Some books could have appeared on more than one order. Hint: We should first identify what orders contain the book with ISBN 0401140733. SELECT CONCAT(lastname, CONCAT(‘,’), firstname)) AS “Customer”, order#, title, isbn FROM customers c NATURAL JOIN orders o NATURAL JOIN orderitems oi NATURAL JOIN books b WHERE order# =ANY (SELECT order# FROM orderitems WHERE isbn = ‘0401140733’) ORDER BY order#; Find out those customers (from any state) who have a total amount due greater than the highest total amount due for the customers living in Florida. In other words, the book store manager wants to know who owes more than all of those customers who live in FL. SELECT CONCAT(lastname, CONCAT(‘,’), firstname)) AS “Customer”, state, order#, SUM(quantitiy\*paideach) AS “TOTAL AMOUNT DUE”, MAX(quantity\*paideach) AS “Max Due of All of f1” FROM customers c NATURAL JOIN orders o NATURAL JOIN orderitems oi GROUP BY CONCAT(lastname, CONCAT(‘,’), firstname)), state, order# HAVING SUM(quantity\*paideach>ALL (SELECT MAX(quantity\*paideach) FROM customers c NATURAL JOIN orders o NATURAL JOIN orderitesm oi WHERE stat=’FL’); Identifies the book(s) with a highest retail value in its own category. SELECT b.title, b.retail, t.category, t.catmaxretail FROM books b, (SELECT category, MAX(retail)catmaxretail FROM books GROUP BY category)t WHERE b.category = t.category and b.retail = t.catmaxretai ORDER BY t.category;