-- 1. Prepare a list of offices sorted by country, state, city.

SELECT country, state, city

FROM classicmodels.offices;

-- 2. How many employees are there in the company?

SELECT count(employeenumber)

FROM classicmodels.employees;

-- 3. What is the total of payments received?

SELECT sum(amount)

FROM classicmodels.payments;

-- 4. List the product lines that contain 'Cars'.

SELECT productLine

FROM classicmodels.productlines

Where productLine like '%Cars%';

-- 5. Report total payments for October 28, 2004.

SELECT sum(amount)

FROM classicmodels.payments

WHERE paymentDate = '2004-10-28';

-- 6. Report those payments greater than $100,000.

SELECT \*

FROM classicmodels.payments

WHERE amount < 100000;

-- 7. List the products in each product line.

SELECT \*

FROM classicmodels.products

ORDER BY productLine;

-- 8. How many products in each product line?

SELECT productLine, count(productCode) as product\_quantity

FROM classicmodels.products

GROUP BY productLine

ORDER BY product\_quantity DESC;

-- 9. What is the minimum payment received?

SELECT min(amount)

FROM classicmodels.payments;

-- 10. List all payments greater than twice the average payment.

SELECT amount

FROM classicmodels.payments

WHERE amount > 2 \* (select avg(amount) from classicmodels.payments);

-- 11. What is the average percentage markup of the MSRP on buyPrice?

SELECT (avg(MSRP) - avg(buyPrice))/avg(buyPrice)

FROM classicmodels.products;

-- 12. How many distinct products does ClassicModels sell?

SELECT count(productCode)

FROM classicmodels.products;

-- 13. Report the name and city of customers who don't have sales representatives?

SELECT CustomerName, city

FROM classicmodels.customers

WHERE salesRepEmployeeNumber IS NULL;

-- 14. What are the names of executives with VP or Manager in their title? Use the CONCAT function to combine the employee's first name and last name into a single field for reporting.

SELECT concat(firstName, lastName)

FROM classicmodels.employees

WHERE jobTitle like '%VP%' or jobTitle like '%Manager%';

-- 15. Which orders have a value greater than $5,000?

SELECT \*

FROM classicmodels.payments

WHERE amount > 5000

ORDER BY amount;

having total\_value>5000;

-- One to many relationship

-- 1. Report the account representative for each customer.

select customername, concat(firstname,' ',lastname) as salesRep\_name from customers c, employees e

where c.salesrepemployeenumber=e.employeenumber;

-- 2. Report total payments for Atelier graphique.

select customername, sum(amount) as total\_payment from payments p, customers c

where c.customernumber=p.customernumber

and customername='Atelier graphique';

-- 3. Report the total payments by date

select sum(amount) as total\_payment, paymentdate from payments

group by paymentdate;

-- 4. Report the products that have not been sold.

select productname from products p

where p.productcode not in (select distinct productcode from orderdetails);

-- 5. List the amount paid by each customer.

select customername, sum(amount) as total\_amount from customers c, payments p

where p.customernumber=c.customernumber

group by customername;

-- 6. How many orders have been placed by Herkku Gifts?

select customername, count(ordernumber) as total\_orders from customers c, orders o

where o.customernumber=c.customernumber

and customername='Herkku Gifts';

-- 7. Who are the employees in Boston?

select employeenumber, concat(firstname,' ',lastname) as emp\_name, city from offices o, employees e

where e.officecode=o.officecode

and city='Boston';

-- 8. Report those payments greater than $100,000. Sort the report so the customer who made the highest payment appears first.

select customername, amount from payments p, customers c

where p.customernumber=c.customernumber

and amount>100000

order by amount desc;

-- 9. List the value of 'On Hold' orders.

select o.ordernumber, o.status, sum(priceeach\*quantityordered) as total\_value from orders o, orderdetails d

where o.ordernumber=d.ordernumber

and o.status='On Hold'

group by o.ordernumber;

-- 10. Report the number of orders 'On Hold' for each customer.

select customername, count(ordernumber) as total\_onHold\_orders from customers c, orders o

where c.customernumber=o.customernumber

and o.status='On Hold'

group by c.customernumber;

-- Many to many relationship

-- 1. List products sold by order date.

select productname, orderdate from orders o, orderdetails d, products p

where p.productcode=d.productcode

and o.ordernumber=d.ordernumber

order by orderdate;

-- 2. List the order dates in descending order for orders for the 1940 Ford Pickup Truck.

select orderdate from orders o, orderdetails d, products p

where p.productcode=d.productcode

and o.ordernumber=d.ordernumber

and productname='1940 Ford Pickup Truck'

order by orderdate desc;

-- 3. List the names of customers and their corresponding order number where a particular order from that customer has a value greater than $25,000?

select customername, o.ordernumber, sum(priceeach\*quantityordered) as total\_value from customers c, orders o, orderdetails d

where c.customernumber=o.customernumber

and o.ordernumber=d.ordernumber

group by o.ordernumber

having total\_value>25000;

-- 4. Are there any products that appear on all orders?

select distinct productname from products p, orderdetails o

where p.productcode=o.productcode

group by o.ordernumber

having count(distinct o.ordernumber) =count(distinct o.ordernumber,o.productcode);

-- 5. List the names of products sold at less than 80% of the MSRP.

select distinct productname from orderdetails o, products p

where o.productcode=p.productcode

and 1.0\*priceeach/msrp<0.8;

-- 6. Reports those products that have been sold with a markup of 100% or more (i.e., the priceEach is at least twice the buyPrice)

select distinct productname from orderdetails o, products p

where o.productcode=p.productcode

and priceeach>buyprice\*2;

-- 7. List the products ordered on a Monday.

select productname, orderdate from orders o, orderdetails d, products p

where p.productcode=d.productcode

and o.ordernumber=d.ordernumber

and weekday(orderdate)=0

order by orderdate;

-- 8. What is the quantity on hand for products listed on 'On Hold' orders?

select productname, quantityinstock from orders o, orderdetails d, products p

where p.productcode=d.productcode

and o.ordernumber=d.ordernumber

and o.status='On Hold';

-- Regular expressions

-- 1. Find products containing the name 'Ford'.

select productname from products

where productname like '%Ford%';

-- 2. List products ending in 'ship'.

select productname from products

where productname like '%ship';

-- 3. Report the number of customers in Denmark, Norway, and Sweden.

select country, count(distinct customernumber) as total\_customer from customers

where country in ('Denmark', 'Norway', 'Sweden')

group by country;

-- 4. What are the products with a product code in the range S700\_1000 to S700\_1499?

select productname, productcode from products

where productcode regexp 'S700\_1[0-4][0-9][0-9]';

-- 5. Which customers have a digit in their name?

select customername from customers

where customername regexp '[0-9]';

-- 6. List the names of employees called Dianne or Diane.

select firstname, lastname from employees

where firstname in ('Dianne', 'Diane')

or lastname in ('Dianne', 'Diane');

-- 7. List the products containing ship or boat in their product name.

select productname from products

where productname like '%ship%'

or productname like '%boat%';

-- 8. List the products with a product code beginning with S700.

select productname, productcode from products

where productcode like 'S700%';

-- 9. List the names of employees called Larry or Barry.

select firstname, lastname from employees

where firstname in ('Larry', 'Barry')

or lastname in ('Larry', 'Barry');

-- 10. List the names of employees with non-alphabetic characters in their names.

select firstname, lastname from employees

where firstname like '%[^a-zA-Z]%'

or lastname like '%[^a-zA-Z]%';

-- 11. List the vendors whose name ends in Diecast

select distinct productvendor from products

where productvendor like '%Diecast';

-- General queries

-- 1. Who is at the top of the organization (i.e., reports to no one).

select employeenumber, firstname, lastname from employees

where reportsto is null;

-- 2. Who reports to William Patterson?

select employeenumber, firstname, lastname from employees

where reportsto=(select employeenumber from employees where firstname='William' and lastname='Patterson');

-- 3. List all the products purchased by Herkku Gifts.

select distinct productname, customername from products p, orderdetails d, orders o, customers c

where c.customernumber=o.customernumber

and o.ordernumber=d.ordernumber

and p.productcode=d.productcode

and c.customername='Herkku Gifts' ;

-- 4. Compute the commission for each sales representative, assuming the commission is 5% of the value of an order. Sort by employee last name and first name.

select lastname, firstname, 0.05\*sum(priceeach\*quantityordered) as commission from employees e

left join customers c on e.employeenumber=c.salesrepemployeenumber

left join orders o on o.customernumber=c.customernumber

left join orderdetails d on o.ordernumber=d.ordernumber

group by salesrepemployeenumber

order by lastname, firstname;

-- 5. What is the difference in days between the most recent and oldest order date in the Orders file?

select datediff(max(orderdate), min(orderdate)) as date\_difference from orders;

-- 6. Compute the average time between order date and ship date for each customer ordered by the largest difference.

select customername, avg(datediff(shippeddate, orderdate)) as date\_difference from customers c, orders o

where o.customernumber=c.customernumber

group by c.customernumber

order by date\_difference desc;

-- 7. What is the value of orders shipped in August 2004? (Hint).

select left(shippeddate, 7) as ship\_month, sum(priceeach\*quantityordered) as value\_aug2004 from orders o, orderdetails d

where o.ordernumber=d.ordernumber

and left(shippeddate, 7)='2004-08'

group by ship\_month;

-- 8. Compute the total value ordered, total amount paid, and their difference for each customer for orders placed in 2004 and payments received in 2004 (Hint; Create views for the total paid and total ordered).

select \*, total\_value-total\_payment as amount\_difference from (

select customername, sum(priceeach\*quantityordered) as total\_value, sum(amount) as total\_payment from customers c

left join orders o on o.customernumber=c.customernumber

left join orderdetails d on d.ordernumber=o.ordernumber

left join payments p on p.customernumber=c.customernumber

where left(o.orderdate,4)='2004'

and left(p.paymentdate,4)='2004'

group by c.customernumber)t;

-- 9. List the employees who report to those employees who report to Diane Murphy. Use the CONCAT function to combine the employee's first name and last name into a single field for reporting.

select concat(e1.firstname, ' ', e1.lastname)as full\_name from employees e1

join employees e2 on e1.reportsto=e2.employeenumber

join employees e3 on e2.reportsto=e3.employeenumber

where concat(e3.firstname, ' ', e3.lastname)='Diane Murphy';

-- 10. What is the percentage value of each product in inventory sorted by the highest percentage first (Hint: Create a view first).

select productname, quantityinstock, concat(quantityinstock/total\_inv\*100, '%') as percentage\_value

from products p,(select sum(quantityinstock) as total\_inv from products) t

order by percentage\_value desc;

-- 11. Write a function to convert miles per gallon to liters per 100 kilometers.

DELIMITER

create function MPG\_to\_LPK(mpg decimal(10, 2), distance decimal(10, 2) )

returns decimal(10, 2)

deterministic

begin

declare liters\_needed decimal(10, 2);

set liters\_needed = ((100 \* 3.785411784) / (mpg \* 1.609344)/ 100) \* distance;

return liters\_needed;

end

select MPG\_to\_LPK(32, 60);

-- 12. Write a procedure to increase the price of a specified product category by a given percentage. You will need to create a product table with appropriate data to test your procedure. Alternatively, load the ClassicModels database on your personal machine so you have complete access. You have to change the DELIMITER prior to creating the procedure.

drop procedure increase\_price

DELIMITER

create procedure increase\_price(in target\_prod\_code varchar(15), in percentage decimal)

begin

update products

set msrp=(1+percentage/100)\*msrp

where productcode=target\_prod\_code;

end

DELIMITER

call increase\_price('test\_10000', 20);

-- 13. What is the value of orders shipped in August 2004? (Hint).

select left(shippeddate, 7) as ship\_month, sum(priceeach\*quantityordered) as value\_aug2004 from orders o, orderdetails d

where o.ordernumber=d.ordernumber

and left(shippeddate, 7)='2004-08'

group by ship\_month;

-- 14. What is the ratio the value of payments made to orders received for each month of 2004. (i.e., divide the value of payments made by the orders received)?

select left(orderdate, 7) as order\_month, sum(priceeach\*quantityordered) as total\_value, sum(amount) as total\_payment, sum(priceeach\*quantityordered)/sum(amount) as ratio

from orders o

join orderdetails d on o.ordernumber=d.ordernumber

join payments p on p.customernumber=o.customernumber

where year(orderdate)=2004

and year(paymentdate)=2004

group by order\_month;

-- 15. What is the difference in the amount received for each month of 2004 compared to 2003?

select t1.months, total\_amount\_2003, total\_amount\_2004, total\_amount\_2003-total\_amount\_2004 as differences

from

(select month(paymentdate) as months, sum(amount) as total\_amount\_2003 from payments

where left(paymentdate, 4)='2003'

group by months)t1,

(select month(paymentdate) as months, sum(amount) as total\_amount\_2004 from payments

where left(paymentdate, 4)='2004'

group by months)t2

where t1.months=t2.months

order by t1.months;

-- 16. Write a procedure to report the amount ordered in a specific month and year for customers containing a specified character string in their name.

drop procedure amount\_ordered

DELIMITER

create procedure amount\_ordered(in months int, in years int, in str varchar(50))

begin

select customername, sum(quantityordered\*priceeach) as total\_amount

from customers c, orders o, orderdetails d

where c.customernumber=o.customernumber

and d.ordernumber=o.ordernumber

and year(orderdate)=years

and month(orderdate)=months

and customername like concat('%', str, '%')

group by c.customernumber;

end

DELIMITER

call amount\_ordered(5, 2003, 'i');

-- 17. Write a procedure to change the credit limit of all customers in a specified country by a specified percentage.

drop procedure change\_creditLimit

DELIMITER

create procedure change\_creditLimit(in cust\_country varchar(50), in percentage decimal)

begin

update customers

set creditLimit=creditLimit\*(1+percentage/100)

where country=cust\_country;

end

DELIMITER

call change\_creditLimit('UK', 20);

-- 18. Basket of goods analysis: A common retail analytics task is to analyze each basket or order to learn what products are often purchased together. Report the names of products that appear in the same order ten or more times.

select distinct case when product\_1>product\_2 then concat(product\_1,' & ',product\_2) else concat(product\_2,' & ',product\_1) end as combinations

from

(select distinct d1.productcode as product\_1, d2.productcode as product\_2 from orderdetails d1, orderdetails d2

where d1.ordernumber=d2.ordernumber

and d1.productcode!=d2.productcode

group by d1.productcode, d2.productcode

having count(\*)>=10)t

group by combinations;

-- 19. ABC reporting: Compute the revenue generated by each customer based on their orders. Also, show each customer's revenue as a percentage of total revenue. Sort by customer name.

/\* select customername, sum(amount) as revenue, sum(amount)/total\_revenue as ratio from payments p, customers c,

(select sum(amount) as total\_revenue from payments) t

where c.customernumber=p.customernumber

group by c.customernumber

order by customername; \*/

select customername, sum(priceeach\*quantityordered) as revenue, sum(priceeach\*quantityordered)/total\_revenue as ratio

from customers c, orders o, orderdetails d, (select sum(priceeach\*quantityordered) as total\_revenue from orderdetails) t

where c.customernumber=o.customernumber

and o.ordernumber=d.ordernumber

group by c.customernumber

order by customername;

-- 20. Compute the profit generated by each customer based on their orders. Also, show each customer's profit as a percentage of total profit. Sort by profit descending.

select customername, sum((priceeach-buyprice)\*quantityordered) as profit, sum((priceeach-buyprice)\*quantityordered)/total\_profit as ratio

from customers c, orders o, orderdetails d, products p, (select sum((priceeach-buyprice)\*quantityordered) as total\_profit from orderdetails d, products p where d.productcode=p.productcode) t

where c.customernumber=o.customernumber

and o.ordernumber=d.ordernumber

and p.productcode=d.productcode

group by c.customernumber

order by profit desc;

-- 21. Compute the revenue generated by each sales representative based on the orders from the customers they serve.

select salesrepemployeenumber, sum(priceeach\*quantityordered) as revenue, sum(priceeach\*quantityordered)/total\_revenue as ratio

from customers c, orders o, orderdetails d, (select sum(priceeach\*quantityordered) as total\_revenue from orderdetails) t

where c.customernumber=o.customernumber

and o.ordernumber=d.ordernumber

group by c.salesrepemployeenumber;

-- 22. Compute the profit generated by each sales representative based on the orders from the customers they serve. Sort by profit generated descending.

select salesrepemployeenumber, sum((priceeach-buyprice)\*quantityordered) as profit, sum((priceeach-buyprice)\*quantityordered)/total\_profit as ratio

from customers c, orders o, orderdetails d, products p, (select sum((priceeach-buyprice)\*quantityordered) as total\_profit from orderdetails d, products p where d.productcode=p.productcode) t

where c.customernumber=o.customernumber

and o.ordernumber=d.ordernumber

and p.productcode=d.productcode

group by c.salesrepemployeenumber

order by profit desc;

-- 23. Compute the revenue generated by each product, sorted by product name.

select productname, sum(priceeach\*quantityordered) as revenue

from orderdetails d, products p

where p.productcode=d.productcode

group by p.productcode

order by productname;

-- 24. Compute the profit generated by each product line, sorted by profit descending.

select productline, sum((priceeach-buyprice)\*quantityordered) as profit

from orderdetails d, products p

where p.productcode=d.productcode

group by productline

order by profit desc;

-- 25. Same as Last Year (SALY) analysis: Compute the ratio for each product of sales for 2003 versus 2004.

select t2003.productcode, sales\_2003, sales\_2004, sales\_2003/sales\_2004 as ratio

from

(select productcode, sum(priceeach\*quantityordered) as sales\_2003 from orders o, orderdetails d

where o.ordernumber=d.ordernumber

and year(orderdate)=2003

group by productcode)t2003,

(select productcode, sum(priceeach\*quantityordered) as sales\_2004 from orders o, orderdetails d

where o.ordernumber=d.ordernumber

and year(orderdate)=2004

group by productcode)t2004

where t2003.productcode=t2004.productcode;

-- 26. Compute the ratio of payments for each customer for 2003 versus 2004.

select t2003.customername, payment\_2003, payment\_2004, payment\_2003/payment\_2004 as ratio

from

(select customername, c.customernumber, sum(amount) as payment\_2003 from customers c, payments p

where c.customernumber=p.customernumber

and year(paymentdate)=2003

group by c.customernumber)t2003,

(select customername, c.customernumber, sum(amount) as payment\_2004 from customers c, payments p

where c.customernumber=p.customernumber

and year(paymentdate)=2004

group by c.customernumber)t2004

where t2003.customernumber=t2004.customernumber;

-- 27. Find the products sold in 2003 but not 2004.

select distinct productname from products p, orderdetails d, orders o

where p.productcode=d.productcode

and o.ordernumber=d.ordernumber

and left(o.orderdate, 4)='2003'

and d.productcode not in (select distinct productcode from orderdetails d, orders o

where o.ordernumber=d.ordernumber

and left(o.orderdate, 4)='2004');

-- 28. Find the customers without payments in 2003.

select customername from customers c

where c.customernumber not in (select customernumber from payments where left(paymentdate, 4)='2003');

-- Correlated subqueries

-- 1. Who reports to Mary Patterson?

select firstname, lastname from employees

where reportsto=(select employeenumber from employees where firstname='Mary' and lastname='Patterson');

-- 2. Which payments in any month and year are more than twice the average for that month and year (i.e. compare all payments in Oct 2004 with the average payment for Oct 2004)? Order the results by the date of the payment. You will need to use the date functions.

select paymentdate, amount

from payments p, (select left(paymentdate, 7) as months, avg(amount) as avg\_amount from payments group by months)t

where left(p.paymentdate, 7)=t.months

and amount>2\*avg\_amount

order by paymentdate;

-- 3. Report for each product, the percentage value of its stock on hand as a percentage of the stock on hand for product line to which it belongs. Order the report by product line and percentage value within product line descending. Show percentages with two decimal places.

select \*, round(quantityInStock/line\_quant\*100, 2) as percentage

from

(select productLine, productCode, quantityInStock, sum(quantityinstock) over (partition by productline) as line\_quant

from products

group by productLine, productCode

order by productLine, productCode)t

order by productLine, percentage desc;

-- 4. For orders containing more than two products, report those products that constitute more than 50% of the value of the order.

select t2.ordernumber, t2.productcode, product\_value, order\_value

from

(select ordernumber, sum(priceeach\*quantityordered) as order\_value from orderdetails

group by ordernumber

having count(distinct productcode)>2)t1,

(select ordernumber, productcode, sum(priceeach\*quantityordered) as product\_value

from orderdetails

group by ordernumber, productcode)t2

where t1.ordernumber=t2.ordernumber

and product\_value>0.5\*order\_value;

-- Spatial data

-- The Offices and Customers tables contain the latitude and longitude of each office and customer in officeLocation and customerLocation, respectively, in POINT format. Conventionally, latitude and longitude and reported as a pair of points, with latitude first.

-- 1. Which customers are in the Southern Hemisphere?

select customername, country from customers

where ST\_X(customerLocation)<0;

-- 2. Which US customers are south west of the New York office?

select customername, country, state, city from customers

where ST\_X(customerlocation)<(select ST\_X(officelocation) from offices where city='NYC')

and ST\_Y(customerlocation) not between (select ST\_X(officelocation) from offices where city='NYC') and (select ST\_X(officelocation)+180 from offices where city='NYC');

-- 5. Who is the northernmost customer?

select customername, country from customers

order by ST\_X(customerlocation) desc

limit 1;