Query the two cities in **STATION** with the shortest and longest CITY names, as well as their respective lengths (i.e.: number of characters in the name). If there is more than one smallest or largest city, choose the one that comes first when ordered alphabetically.

**SELECT City, LENGTH(City)**

**FROM (SELECT City**

**FROM Station**

**ORDER BY LENGTH(City), City)**

**WHERE ROWNUM = 1;**

**SELECT City, LENGTH(City)**

**FROM (SELECT City**

**FROM Station**

**ORDER BY LENGTH(City) DESC, City)**

**WHERE ROWNUM = 1;**

Query the list of CITY names starting with vowels (i.e., a, e, i, o, or u) from **STATION**. Your result cannot contain duplicates.

**select distinct city from station where lower(substr(city,1,1)) in ('a','e','i','o','u');**

Query the list of CITY names ending with vowels (a, e, i, o, u) from **STATION**. Your result cannot contain duplicates.

**SELECT DISTINCT CITY FROM STATION WHERE LOWER(SUBSTR(CITY,LENGTH(CITY),1)) IN ('a','e','i','o','u');**

Query the Name of any student in **STUDENTS** who scored higher than  Marks. Order your output by the last three characters of each name. If two or more students both have names ending in the same last three characters (i.e.: Bobby, Robby, etc.), secondary sort them by ascending ID.

**SELECT NAME FROM STUDENTS WHERE MARKS > 75 ORDER BY RIGHT(NAME, 3), ID ASC;**

1. Query an *alphabetically ordered* list of all names in **OCCUPATIONS**, immediately followed by the first letter of each profession as a parenthetical (i.e.: enclosed in parentheses). For example: AnActorName(A), ADoctorName(D), AProfessorName(P), and ASingerName(S).
2. Query the number of ocurrences of each occupation in **OCCUPATIONS**. Sort the occurrences in *ascending order*, and output them in the following format:
3. There are a total of [occupation\_count] [occupation]s.

where [occupation\_count] is the number of occurrences of an occupation in **OCCUPATIONS** and [occupation] is the *lowercase* occupation name. If more than one *Occupation* has the same [occupation\_count], they should be ordered alphabetically.

**select concat(name, '(', substring(occupation, 1, 1), ')') from occupations order by name asc;**

**select concat("There are a total of ", cast(count(\*) as char), " ", lower(occupation), "s.") from occupations group by occupation order by count(\*) asc;**

**WEEK 5 SQL querys**

**-- write SQL statement to print snum, sname for all suppliers**

**SELECT snum, sname**

**FROM suppliers;**

**--SQl statement to print all the field from table Parts**

**SELECT \* from parts;**

**-- SQl statement to print names of the suppliers who are from Paris**

**SELECT sname FROM suppliers WHERE city='Paris';**

**-- SQl statement to retrieve pnum for those parts that are supplied to project 'J2'. Sort the part names in ascending order**

**SELECT pnum**

**FROM shipments**

**WHERE jnum = 'J2'**

**ORDER BY pnum;**

**-- SQl statement to print pnum of those parts which do not supply to project 'J2'**

**-- answer 1 (incorrect - why?)**

**SELECT pnum**

**FROM shipments**

**WHERE jnum != 'J2';**

**-- answer 2 (incorrect)**

**SELECT pnum**

**FROM shipments**

**WHERE not jnum = 'J2';**

**-- answer 3 (correct)**

**SELECT pnum**

**FROM parts**

**EXCEPT**

**SELECT pnum**

**FROM shipments**

**WHERE jnum = 'J2';**

**-- SQl statement to print a list of parts coming from Paris and supplying to project 'J2'**

**-- expected answer: P2 and P5**

**SELECT pnum from parts where city = 'Paris'**

**INTERSECT**

**SELECT pnum FROM shipments WHERE jnum = 'J2'**

**ORDER BY pnum;**

**-- SQl statement to print all part numbers (pnum) and part names (pname) of those parts that are carried in one of the following colors:**

**-- red, yellow, or green**

**SELECT pnum, pname**

**FROM parts**

**WHERE color = 'Red' OR color='Green' OR color = 'Yellow';**

**-- use UNION for the same query**

**SELECT pnum, pname**

**FROM parts**

**WHERE color = 'Red'**

**UNION**

**(SELECT pnum, pname**

**FROM parts**

**WHERE color = 'Yellow')**

**UNION**

**(SELECT pnum, pname**

**FROM parts**

**WHERE color = 'Green')**

**ORDER BY pnum DESC;**

**-- Retrieve the total number of suppliers in the database**

**-- incorrect - why?**

**SELECT SUM(\*)**

**FROM suppliers;**

**-- average status of supplier**

**-- should print 20.833333333**

**SELECT AVG(status)**

**FROM suppliers;**

**WEEK 6 SQL Queries**

**SELECT DISTINCT sname**

**FROM suppliers NATURAL JOIN shipments**

**WHERE pnum = 'P2';**

**-- names of suppliers who supply at least one red part**

**SELECT DISTINCT sname**

**FROM suppliers NATURAL JOIN shipments INNER JOIN parts USING(pnum)**

**WHERE parts.color='Red';**

**--retrieve names and numbers of all suppliers who supply either a red part or a blue part**

**SELECT DISTINCT sname, snum**

**FROM suppliers NATURAL JOIN shipments INNER JOIN parts USING (pnum)**

**WHERE parts.color = 'Blue' or parts.color = 'Red'**

**ORDER BY snum;**

**--retrieve names and numbers of all suppliers who supply both red and blue parts**

**SELECT DISTINCT sname, snum**

**FROM suppliers NATURAL JOIN shipments INNER JOIN parts USING (pnum)**

**WHERE parts.color = 'Blue' AND parts.color = 'Red'**

**ORDER BY snum;**

**-- parts cannot be both red and blue at the same time**

**--new solution**

**(SELECT sname, snum**

**FROM parts**

**JOIN shipments USING (pnum)**

**JOIN suppliers USING (snum)**

**WHERE color = 'Red')**

**INTERSECT**

**(SELECT sname, snum**

**FROM parts**

**JOIN shipments USING (pnum)**

**JOIN suppliers USING (snum)**

**WHERE color = 'Blue');**

**-- self joining of tables**

**-- pairs of those suppliers that live in the same city**

**SELECT DISTINCT suppA.snum AS suppAsnum, suppB.snum AS suppBsnum**

**FROM suppliers AS suppA INNER JOIN suppliers AS suppB**

**ON suppA.city = suppB.city**

**WHERE suppA.snum != suppB.snum AND suppA.snum < suppB.snum**

**ORDER BY suppA.snum;**

**--name, number, color of all parts that are supplied by supplier 'S3'**

**SELECT parts.pname, parts.pnum, parts.color**

**FROM shipments NATURAL JOIN parts**

**WHERE shipments.snum = 'S3';**

**----name, number, color of all parts that are supplied by supplier 'S3'**

**--incorrect answer. Why?**

**SELECT DISTINCT parts.pname, parts.pnum, parts.color**

**FROM shipments NATURAL JOIN parts**

**WHERE shipments.snum != 'S3';**

**--correct answer**

**SELECT parts.pname, parts.pnum, parts.color**

**FROM parts**

**EXCEPT**

**(SELECT parts.pname, parts.pnum, parts.color**

**FROM shipments NATURAL JOIN parts**

**WHERE shipments.snum = 'S3');**