Task 5-1 *Modification Operations in SQL*

The relations Customer, Staff, Sales, Inventory, and Order Items are given as a data model for a furniture sales database:

Customer (custid, custname, address, city, postalcode)

Staff (staffid, lastname, firstname, department, supervisor, salary)

Sales (orderid, orderdate, staffid, custid)

Inventory (itemid, itemdesc, stock, stockloc, price)

Order Items (orderid, itemid, quantity)

Formulate the following modification operations in SQL. Assume that the foreign keys are defined with on delete cascade. Provide the necessary modification commands for each sub-task in an appropriate order. Assume that the database corresponds to the example state at the end of the exercise sheet.

1. 12 pieces of furniture with the description ’FATBOY BEANBAG’ are delivered to the Hamburg warehouse. Add a corresponding entry with item number ’424242’ and a price of 90.00 to the Inventory table.

INSERT INTO Inventory (item\_id, item\_desc, stock, stock\_loc, price)

VALUES ('424242', 'FATBOY BEANBAG', 12, 'Hamburg warehouse', 90.00);

1. The company’s managers (i.e., people with supervisor = 0) treat themselves to a salary increase. Double their salary.

UPDATE Staff

SET salary = salary \* 2

WHERE supervisor = 0;

1. The manager with staff id=8 is dismissed for early retirement due to embezzlement of company funds. Consequently, their profile is also removed from the database. Additionally, the employee with staff id=12 is transferred to Munich, their salary is increased by 10,000, and they are promoted to the new manager (i.e., supervisor=0). They also take over all subordinates and sales from their predecessor. Formulate the corresponding Insert, Update, and Delete operations to update the database. Ensure the correct sequence.

UPDATE Staff

SET supervisor = 12

WHERE supervisor = 8

UPDATE Sales

SET staff\_id = 12

WHERE staff\_id = 8

UPDATE Staff

SET department = 'Munich', salary = salary + 10000, supervisor = 0

WHERE staff\_id = 12

DELETE FROM Staff

WHERE staff\_id = 8;

Task 5-2 *Join Operations in SQL*

In the following, the symbol ⋊ denotes a Left Outer Join and the symbol ⋉ denotes a Right Outer Join. Both symbols can be used to modify Theta, Equi, or Natural Joins. For example, ⋉ represents a Right Outer

*T*1*.nr*=*T*2*.nr*

Equi Join. The two relations T1(nr, name) and T2 (nr, value) with the following values are given:

|  |  |
| --- | --- |
| nr | value |
| 4 | w |
| 5 | x |
| 5 | y |
| 7 | z |

|  |  |
| --- | --- |
| nr | name |
| 1 | a |
| 4 | b |
| 5 | c |

T1T2

Formulate the following queries in SQL and provide the result relation. You may use Join operators, but it is not required.

1. *T*1×*T*2

SELECT T1.nr, T1.name, T2.nr AS nr\_T2, T2.value

FROM T1, T2;

|  |  |  |  |
| --- | --- | --- | --- |
| nr | name | Nr\_TR2 | T2.value |
| 1 | a | 4 | w |
| 1 | a | 5 | x |
| 1 | a | 5 | y |
| 1 | a | 7 | z |
| 4 | b | 4 | w |
| 4 | b | 5 | x |
| 4 | b | 5 | y |
| 4 | b | 7 | z |
| 5 | c | 4 | w |
| 5 | c | 5 | x |
| 5 | c | 5 | y |
| 5 | c | 7 | z |

1. 

SELECT T1.nr, T1.name, T2.value

FROM T1

JOIN T2 ON T1.nr = T2.nr;

|  |  |  |
| --- | --- | --- |
| nr | name | value |
| 4 | b | w |
| 5 | c | x |
| 5 | c | y |

1. *T*1*▷◁T*2

SELECT T1.nr, T1.name, T2.value

FROM T1

NATURAL JOIN T2;

|  |  |  |
| --- | --- | --- |
| nr | name | value |
| 4 | b | w |
| 5 | c | x |
| 5 | c | y |

1. *T*1 ⋉ *T*2

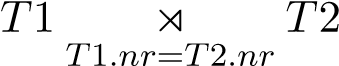
*T*1*.nr*=*T*2*.nr*

SELECT T1.nr, T1.name, T2.value

FROM T1

RIGHT JOIN T2 ON T1.nr = T2.nr;

|  |  |  |
| --- | --- | --- |
| nr | name | value |
| 4 | b | w |
| 5 | c | x |
| 5 | c | y |
| NULL | NULL | z |

1. 

SELECT T1.nr, T1.name, T2.value

FROM T1

LEFT JOIN T2 ON T1.nr = T2.nr;

|  |  |  |
| --- | --- | --- |
| nr | name | value |
| 1 | a | NULL |
| 4 | b | w |
| 5 | c | x |
| 5 | c | y |

Task 5-3 *Queries in SQL (I:a-f , II: g-j)*

This task refers to the furniture database schema known from previous exercise sheets. Formulate the following queries in SQL. An SQL interface for this schema is also available on the lecture website.

1. Find the item number, item description, storage location, and stock of all items stored in Hamburg or Munich.

SELECT item\_id, item\_desc, stock\_loc, stock

FROM Inventory

WHERE stock\_loc IN ('Hamburg', 'Munich');

1. For all orders of the item with number 203333 stored in the table Commisions, find the order number, item number, ordered quantity, and all stock levels and storage locations where a sufficient quantity of item 203333 is available.

SELECT O.order\_id, O.item\_id, O.quantity, I.stock, I.stock\_loc

FROM Order\_Items O

JOIN Inventory I ON O.item\_id = I.item\_id

WHERE O.item\_id = 203333 AND I.stock >= O.quantity;

1. Find the numbers of those customers who have not placed any orders.

SELECT cust\_id

FROM Customer WHERE cust\_id NOT IN (SELECT cust\_id FROM Sales);

1. Find the different storage locations of items in the Inventory table that have a stock of at least 8 units.

SELECT DISTINCT stock\_loc

FROM Inventory

WHERE stock >= 8;

1. Find the first name and last name of those employees who have processed one or more sales to customers from Stuttgart.

SELECT DISTINCT S.first\_name, S.last\_name

FROM Staff S

JOIN Sales Sa ON S.staff\_id = Sa.staff\_id

JOIN Customer C ON Sa.cust\_id = C.cust\_id

WHERE C.city = 'Stuttgart';

1. Find the last name, first name, deployment location, and salary of all employees stored in the Personal table. The result should be sorted lexicographically by deployment location. For the same deployment location, it should be sorted in descending order by salary.

SELECT last\_name, first\_name, department AS deployment\_location, salary

FROM Staff

ORDER BY department ASC, salary DESC;

1. Find the employee numbers and salaries of employees with the minimum or maximum salary.

SELECT staff\_id, salary

FROM Staff

WHERE salary = (SELECT MIN(salary) FROM Staff) OR salary = (SELECT MAX(salary) FROM Staff);

1. Find the number of employees deployed to each job site.

SELECT department AS job\_site, COUNT(\*) AS num\_employees

FROM Staff

GROUP BY department;

1. What is the average number of employees across all deployment locations?

SELECT AVG(emp\_count) AS average\_num\_employees

FROM (

SELECT department,

COUNT(\*) AS emp\_count

FROM Staff

GROUP BY department

) AS dept\_counts;

1. Find the item number and the total stock for each item in the storage locations that stock the item. Only items with a total stock of more than 10 units should be displayed.

SELECT item\_id, SUM(stock) AS total\_stock

FROM Inventory

GROUP BY item\_id

HAVING SUM(stock) > 10;

Task 5-4 *Queries in SQL (I:a-c, II:d)*

Given are the relations Professors, Students, Courses, Prerequisites, attend, and Exams as a data model for a university database:

Professors (persNr, name, office)

Students (matrNr, name, semester)

Courses (courseNr, title, ects, taughtBy[Professors] Prerequisites (prerequisite[Courses], course[Courses]) attend (matrNr[Students],courseNr[Courses])

Exams (matrNr[Students],courseNr[Courses], grade)

Formulate the following queries for the above database schema in SQL.

*Note:* At the end of the exercise sheet, you will find a sample instance of the relation schema to aid understanding. Ensure that your query provides the correct result for any possible instance of the database, not just for the given sample instance.

1. Determine the average grade (rounded to three decimal places) for each course (to display: course number and title) over all exams conducted in that subject.

*Rounding*: To round, use the function ROUND. For example, the result of the calculation in the following query is rounded to three decimal places:

SELECT ROUND( menge\* gewicht / 1000 ,3) FROM LTP NATURAL JOIN T

SELECT Courses.courseNr, Courses.title, ROUND(AVG(Exams.grade), 3) AS avg\_grade

FROM Courses

JOIN Exams ON Courses.courseNr = Exams.courseNr GROUP BY Courses.courseNr, Courses.title;

1. Determine the personal number and name of each professor, along with the number of distinct courses they teach. Professors without courses should be listed with a course count of 0. Sort the results in descending order by course count.

SELECT Professors.persNr, Professors.name, COUNT(DISTINCT Courses.courseNr) AS course\_count

FROM Professors

LEFT JOIN Courses ON Professors.persNr = Courses.taughtBy

GROUP BY Professors.persNr, Professors.name ORDER BY course\_count DESC;

1. Determine, for each student (display: matriculation number and name), how many other students from a higher semester they know (i.e., they attend at least one course together). Show only students who know more than two other students from a higher semester.

SELECT S1.matrNr, S1.name, COUNT(DISTINCT S2.matrNr) AS higher\_semester\_known

FROM Students S1

JOIN attend A1 ON S1.matrNr = A1.matrNr

JOIN attend A2 ON A1.courseNr = A2.courseNr

JOIN Students S2 ON A2.matrNr = S2.matrNr AND S2.semester > S1.semester

GROUP BY S1.matrNr, S1.name

HAVING higher\_semester\_known > 2;

1. Determine the professors (display: staff number and name) for whom at least two students attend at least three of their courses.

SELECT Professors.persNr, Professors.name

FROM Professors

JOIN Courses ON Professors.persNr = Courses.taughtBy

JOIN attend ON Courses.courseNr = attend.courseNr

GROUP BY Professors.persNr, Professors.name, attend.matrNr

HAVING COUNT(DISTINCT Courses.courseNr) >= 3

GROUP BY Professors.persNr, Professors.name

HAVING COUNT(DISTINCT attend.matrNr) >= 2;

*Tip:* Create suitable views to make the query clearer. Use the syntax covered in the lecture!

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Professoren** | | |  | **Pruefungen** | | |
| **persNr** | **name** | **raum** | **matrNr** | **vorlNr** | **note** |
| 1 | Thomas Seidl | 152 | 106810 | 1 | 1,7 |
| 2 | Peer Kröger | F 108 |  | 106810 | 2 | 1,3 |
| 3 | Matthias Schubert | F 107 |  | 106810 | 4 | 2,7 |
| 4 | Volker Tresp | F 111 |  | 106810 | 6 | 2,0 |
| 5 | Hans-Peter Kriegel | F 112 |  | 106810 | 7 | 1,7 |
| 6 | Christian Böhm | 158 |  | 106810 | 9 | 1.0 |
| … | | |  | … | | |
| 17 | Volker Heun | 303 |  | 107245 | 54 | 2,7 |
| 18 | Caroline Friedel | 402 |  | 107279 | 1 | 2,0 |
| 19 | Ralf Zimmer | 401 |  | 107279 | 53 | 2,3 |
| 20 | Heinrich Hußmann | 436 |  | 107279 | 55 | 3,7 |
| 21 | Andreas Butz | 433 |  | 107279 | 4 | 1,3 |
| 22 | Albrecht Schmidt | 442A |  | 107279 | 6 | 4,0 |
| **Studenten** | | |  | | **hoeren** | |
| **matrNr** | **name** | **semester** | **matrNr** | **vorlNr** |
| 106810 Fritz Müller 7  106813 Stefan Schmidt 6   1. Anke Maier 6 2. Julia Petersen 6   106841 Franziska Goldschmidt 5  106844 Sven Bayer 5  …  107245 Lea Freud 2  107279 Katja Richter 2  107290 Wolfgang Jager 1  107375 Mandy Theissen 1  107486 Manuel Ackerman 1  107499 Barbara Gottschalk 1 | | | 106810 51  106810 50  106810 39  106810 49  106810 29  106810 48  …  107486 1  107486 50  107486 55  107486 7  107486 6  107499 2 | |
| **Vorlesungen** | | | |  | **Voraussetzungen** | |
| **vorlNr** | **titel** | **ects** | **gelesenVon** | **vorausgesetzt** | **vorlesung** |
| 1. Einführung in die Programmierung 9 2 2. Digitale Medien 6 20 3. Einführung in die Bioinformatik I 6 19 4. Betriebssysteme 6 9 5. Grundlagen der Analysis 6 16 6. Softwaretechnik 6 11   …   1. Systems Biomedicine 6 18 2. Strukturbioinformatik 6 19 3. Rechnernetze und Verteilte Systeme 6 7 4. Formale Sprachen und Komplexität 6 16 5. Programmierung und Modellierung 6 16 6. Algorithmen und Datenstrukturen 6 1 | | | | 52 30  1 6  53 8  1 9   1. 9 2. 9   …  4 31  55 34  7 34  34 35  34 36  6 38 | |

**Examplary manifestation for the furniture sales data base**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **custom\_id** | **custom\_name** | **address** | **city** | **zip** |
| 'GK4441' | 'Müller & Partner' | 'Schillerstr. 56' | 'Stuttgart' | '74598' |
| 'GK4442' | 'Stadtverwaltung Landshut' | 'Am Rathausplatz' | 'Landshut' | '85321' |
| 'GK4477' | 'Betten Kaiser' | 'Industriepark' | 'Goettingen' | '35134' |
| 'GK4489' | 'Stadtverwaltung Köln' | 'Am Dom 8' | 'Koeln' | '50987' |
| 'EK5689' | 'Paula Peterson' | 'Maistr. 46' | 'Muenchen' | '84977' |
| 'EK5558' | 'Manfred Keller GmbH' | 'Mozartstr. 45' | 'Stuttgart' | '74391' |
| 'GK4333' | 'Planungsbüro Bertoldt' | 'Bauhofstr. 87' | 'Duisburg' | '49155' |
| 'GK4688' | 'Ulrich & Co.' | 'Rosenheimer Str. 234' | 'Muenchen' | '81377' |
| 'EK8992' | 'Peter Hauser' | 'Am Bismarkturm 2' | 'Stuttgart' | '76232' |
| 'EK8993' | 'Miriam Zechmeister KG' | 'Gänsemarkt 36' | 'Hamburg' | '21357' |
| 'GK4443' | 'Freiling & Partner' | 'Engerser Landstr. 17' | 'Koblenz' | '54621' |
| 'GK4490' | 'Einrichtungshaus Röder' | 'Rheinlanddamm 199' | 'Bonn' | '50321' |
| 'GK4491' | 'Stuhl Kaiser' | 'Im Tal 4' | 'Muenchen' | '80432' |
| 'GK4492' | 'Planmöbel GmbH' | 'Am Stockborn 5' | 'Frankfurt' | '61234' |
| 'GK4493' | 'Systemmöbel Wisst & Co.' | 'Greinstr. 2' | 'Koeln' | '53261' |
| 'GK0310' | 'Sitzmöbel Engels' | 'Karl-Marx-Str. 4' | 'Cottbus' | '03096' |

**Customer**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **pers\_id** | **last name** | **given name** | **job site** | **supervisor** | **salary** |
| 1 | 'Winter' | 'Margot' | 'Muenchen' | 8 | 4200 |
| 2 | 'Roser' | 'Michael' | 'Hamburg' | 0 | 5200 |
| 3 | 'Thomas' | 'Fred' | 'Hamburg' | 2 | 4000 |
| 4 | 'Scholl' | 'Friedrich' | 'Muenchen' | 8 | 5300 |
| 5 | 'Hartinger' | 'Roswita' | 'Muenchen' | 8 | 4800 |
| 6 | 'Reitzig' | 'Hans-Peter' | 'Frankfurt' | 7 | 5550 |
| 7 | 'Moll' | 'Rolf' | 'Frankfurt' | 0 | 5900 |
| 8 | 'Sandner' | 'Ernst' | 'Muenchen' | 0 | 6300 |
| 9 | 'Starck' | 'Brigitte' | 'Koeln' | 0 | 5300 |
| 10 | 'Anger' | 'Lars' | 'Koeln' | 9 | 5200 |
| 11 | 'Freudenfeld' | 'Burghard' | 'Muenchen' | 8 | 5500 |
| 12 | 'Neumann' | 'Gabriele' | 'Hamburg' | 2 | 4900 |

**Personal**

|  |  |  |  |
| --- | --- | --- | --- |
| **order \_id** | **order date** | **pers\_id** | **custom\_id** |
| 20002 | '2019-07-23' | 5 | 'GK4688' |
| 20003 | '2019-07-23' | 1 | 'EK8992' |
| 20004 | '2019-07-23' | 8 | 'EK5558' |
| 20005 | '2019-07-24' | 7 | 'GK4491' |
| 20006 | '2019-07-24' | 12 | 'EK8993' |
| 20007 | '2019-07-24' | 12 | 'GK4477' |
| 20008 | '2019-07-24' | 1 | 'GK4442' |
| 20009 | '2019-07-24' | 9 | 'GK4492' |
| 20010 | '2019-07-24' | 11 | 'EK5689' |
| 20011 | '2019-07-24' | 8 | 'GK4441' |

**Sales**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **art\_id** | **art\_descr** | **stock** | **stock location** | **price** |
| 104002 | 'BUECHERREGAL' | 2 | 'Muenchen' | 4100.00 |
| 104002 | 'BUECHERREGAL' | 3 | 'Frankfurt' | 4100.00 |
| 104002 | 'BUECHERREGAL' | 1 | 'Koeln' | 4100.00 |
| 301001 | 'SCHLAFZIMMER' | 6 | 'Muenchen' | 4200.00 |
| 301001 | 'SCHLAFZIMMER' | 20 | 'Koeln' | 4200.00 |
| 104789 | 'BAUERNSCHRANK' | 5 | 'Frankfurt' | 780.00 |
| 201080 | 'COUCH STOCKHOLM' | 1 | 'Frankfurt' | 4800.00 |
| 201080 | 'COUCH STOCKHOLM' | 4 | 'Muenchen' | 4800.00 |
| 201080 | 'COUCH STOCKHOLM' | 2 | 'Hamburg' | 4800.00 |
| 201081 | 'COUCH MIRABELL' | 2 | 'Muenchen' | 2700.00 |
| 203333 | 'COUCH GARNITUR KLASSIKA' | 6 | 'Koeln' | 4300.00 |
| 203333 | 'COUCH GARNITUR KLASSIKA' | 10 | 'Hamburg' | 4300.00 |
| 203333 | 'COUCH GARNITUR KLASSIKA' | 1 | 'Muenchen' | 4300.00 |
| 203333 | 'COUCH GARNITUR KLASSIKA' | 4 | 'Frankfurt' | 4300.00 |
| 201299 | 'COUCH GARNITUR LUXORETT' | 2 | 'Frankfurt' | 10400.00 |
| 104888 | 'KLEIDERSCHRANK' | 5 | 'Muenchen' | 680.00 |
| 104888 | 'KLEIDERSCHRANK' | 12 | 'Frankfurt' | 680.00 |
| 104888 | 'KLEIDERSCHRANK' | 8 | 'Hamburg' | 680.00 |
| 401000 | 'SCHREIBTISCH FUTURA' | 4 | 'Muenchen' | 4200.00 |
| 401000 | 'SCHREIBTISCH FUTURA' | 5 | 'Hamburg' | 4200.00 |
| 401001 | 'DREHSTUHL FUTURA' | 4 | 'Muenchen' | 1800.00 |
| 401001 | 'DREHSTUHL FUTURA' | 5 | 'Hamburg' | 1800.00 |
| 401002 | 'BUEROSYSTEM FUTURA T2' | 3 | 'Muenchen' | 7500.00 |
| 401002 | 'BUEROSYSTEM FUTURA T2' | 5 | 'Hamburg' | 7500.00 |
| 104003 | 'SCHRANKWAND' | 0 | 'Muenchen' | 10500.00 |
| 104003 | 'SCHRANKWAND' | 2 | 'Hamburg' | 10500.00 |

**Inventory**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **order\_id** | **art\_id** | **amount** | | 20002 | 104002 | 2 | | 20002 | 203333 | 3 | | 20002 | 201080 | 1 | | 20003 | 104888 | 5 | | 20003 | 301001 | 1 | | 20004 | 104003 | 1 | | 20004 | 201081 | 2 | | 20004 | 203333 | 3 | | 20005 | 104789 | 1 | | 20006 | 104003 | 1 | | 20006 | 201080 | 2 | | |  |  |  | | --- | --- | --- | | **order\_id** | **art\_id** | **amount** | | 20007 | 401000 | 2 | | 20007 | 401001 | 2 | | 20008 | 401002 | 1 | | 20008 | 401000 | 1 | | 20008 | 203333 | 2 | | 20009 | 301001 | 1 | | 20010 | 301001 | 4 | | 20011 | 301001 | 1 | | 20011 | 401000 | 1 | | 20011 | 401001 | 1 | |

**Commisions**