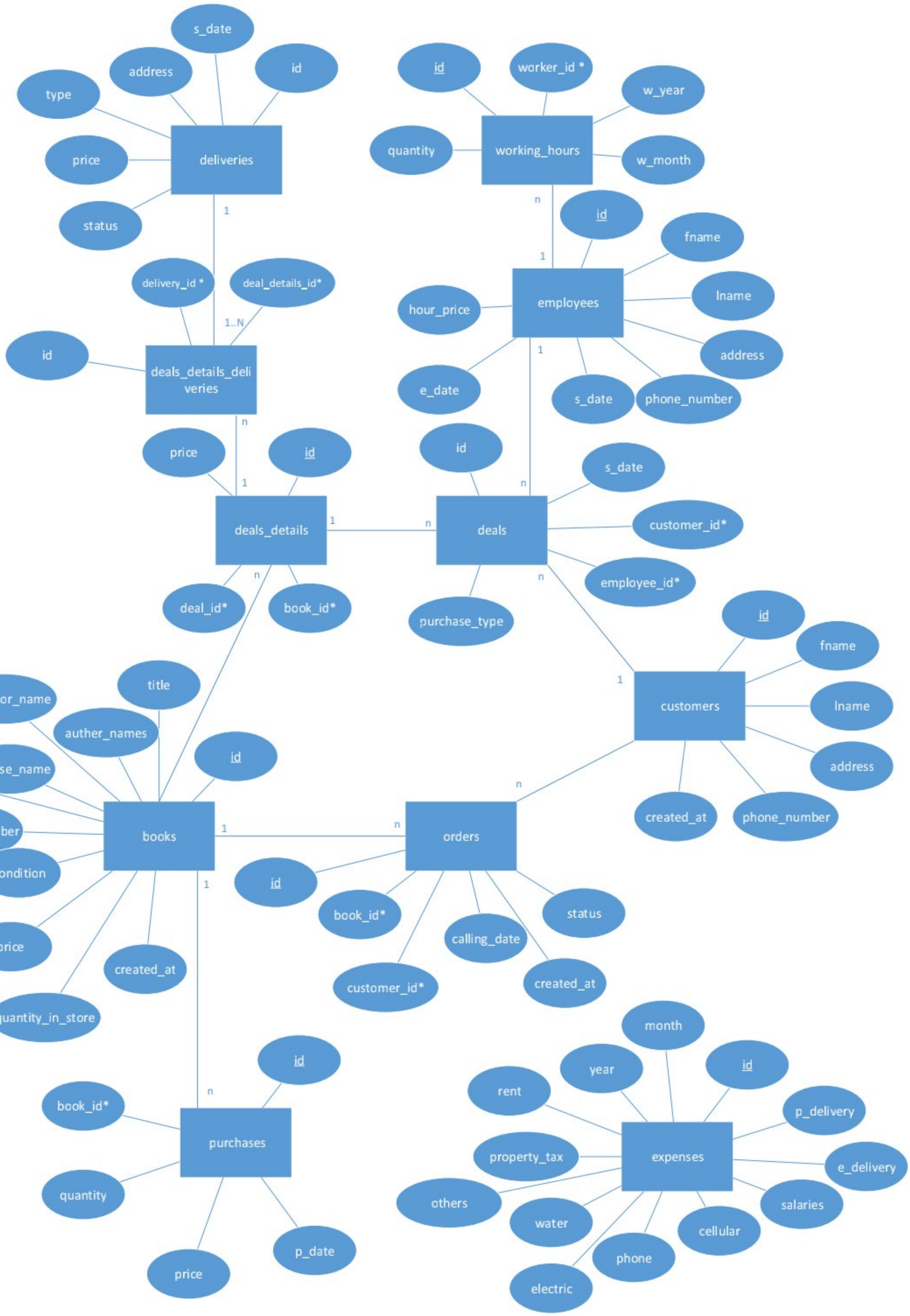
מסמך 1



Scanned with CamScanner

1.Primary key: key

2.Forgein key: key*

1.הקשר בין ספרים ל הזמנות הוא אחד לרבים(ספר הוא לפחות אחד):

אפשר להיות יותר מהזמנה אחת לאותו ספר, ואם יש הזמנה חייב להיות ספר.

2.קשר בין לקוח להזמנות הוא אחד לרבים (הזמנה אחת לפחות):

הלקוח יכול להזמין יותר מהזמנה אחת, ללקוח שכבר קיים הוא בוודאות יש לו הזמנה לפחות.

3.קשר בין לקוח לעסקאות הוא אחד לרבים עסקה יבולה להיות 0:

הלקוח יכול לבצע יותר מעסקה אחת או לא לקנות.

4.קשר בין עסקה לפרטי העסקה הוא אחד לרבים ופרטי העסקה לפחות אחד:

בעסקה יכול להיות קנייה יותר מספר אחד לכן יש יותר מפרטי עסקה אחת אבל לא יכול לקיום עסקה ואין לה פרטים.

5.קשר בין משלוחים לפרטי המשלוח אחד לרבים: יכול להות למשלוח יותר ממקום אחד במקרה שלקוח מפצל את העסקה שלו.

6.קשר בין עובד לעסקה אחד לרבים ולעסקה חייב להיות עובד: עובד יכול לבצע יותר מעסקה אחת והעסקה חייבת להתבצע על ידי עובד אחד בלבד.

7.קשר בין עובד לשעות עבודה הו אחד לרבים: יש לכל עובד רשימת שעות עבודה לכל חודש.

Scanned with CamScanner

מסמך 2

1. לכול ספר יש N רכישות, ולכול רכישה יש ספר אחד
2. לכול לקוח N הזמנות, ולכול הזמנה לקוח אחד וספר אחד
3. לכול לקוח N עסקאות, ולכול עסקה לקוח אחד
4. לכל עסקה יש N ספרים, ולכל ספר נמכר בM עסקאות
5. לכל ספר יש M משלוחים, ובכל משלוח N ספרים
6. לכל עובד יש N עסקאות ולכל עסקה עובד אחד
7. לכל עובד יש N שעות עבודה ולכל שעת עבודה עובד אחד

מסמך 3

Books:

1.{id,title,auther_name->translator_name,release_name,
release_year,page_number,weight,quantity_in_store,condition}

2.{id,title,auther_name,condition->price}

Customers:

1.{id,fname,lname,phone_number->address,created_at}

Deals:

1.{id,s_date,customers->purchase_type,employee_id}

Deals_detail:

1.{id,book_id,deals_id->price}

Deliveries:

1.{id,s_date,address,type->status}

Scanned with CamScanner

Deals_details_deliveries:

1. {id,delivery_id->deal_details_id}

2.{id,deal_details_id->delivery_id}

Emloyees:

{id,fname,lname,phone_number->address,s_date,e_date,hour_price}

Work_hours:

{id,worker_id->w_year,w_month,quantity}

Orders:

1.{id,book_id,costomer_id->calling_date,created_at,status}

Purchase:

{id,book_id->quantity,price,p_date}

Scanned with CamScanner

ממן 4

Books

	Column	Type	Null	Default
Pk	***id***	int(11)	No	
	title	varchar(15)	No	
	auther_name	text	No	
	translator_name	varchar(15)	No	
	release_name	varchar(15)	No	
	release_year	int(11)	No	
	pages_number	int(11)	No	
	weight	int(11)	No	
	condition	varchar(15)	No	
	price	int(11)	No	
	quantity_in_store	int(11)	No	
	created_at	date	No	

Costumers

	Column	Type	Null	Default
Pk	***id***	int(11)	No	
	fname	varchar(11)	No	
	lname	varchar(11)	No	
	address	varchar(11)	No	
	phone_number	varchar(15)	No	
	created_at	date	No	

Deals

	Column	Type	Null	Default
Pk	***id***	int(11)	No	
	s_date	date	No	
	customer_id	int(11)	No	
	purchase_type	varchar(15)	No	
	employee_id	int(11)	No	

Scanned with CamScanner

Deals_details

	Column	Type	Null	Default
Pk	***id***	int(11)	No	
Fk	book_id	int(11)	No	
Fk	deal_id	int(11)	No	
	price	int(11)	No	

Deliveries

	Column	Type	Null	Default
Pk	***id***	int(11)	No	
	s_date	date	No	
	address	text	No	
	type	varchar(15)	No	
	price	int(11)	No	
	status	varchar(15)	No	binding

Deliveries_details

	Column	Type	Null	Default
Pk	***id***	int(11)	No	
Fk	deal_details_id	int(11)	No	
Fk	delivery_id	int(11)	No	

Employees

	Column	Type	Null	Default
Pk	***id***	int(11)	No	
	fname	varchar(11)	No	
	lname	varchar(11)	No	
	address	varchar(11)	No	
	phone_number	varchar(11)	No	
	s_date	date	No	
	e_date	date	Yes	NULL
	hour_price	int(11)	No	

Scanned with CamScanner

Working_hours

	Column	Type	Null	Default
Pk	***id***	int(11)	No	
Fk	worker_id	int(11)	No	
	w_year	int(4)	No	
	w_month	int(2)	No	
	quantity	int(11)	No	

Orders

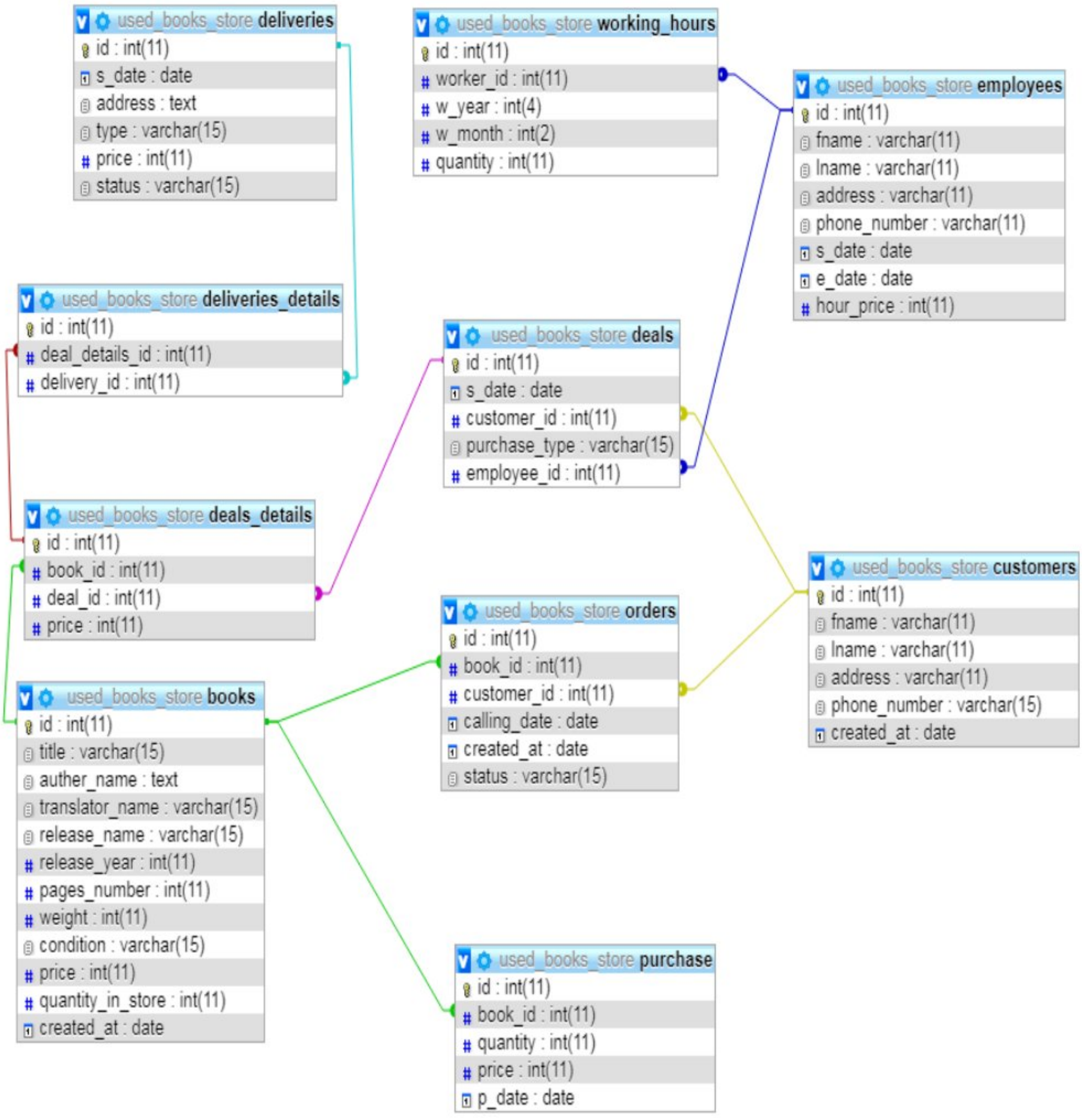
	Column	Type	Null	Default
Pk	***id***	int(11)	No	
Fk	book_id	int(11)	No	
Fk	customer_id	int(11)	No	
	calling_date	date	Yes	NULL
	created_at	date	No	
	status	varchar(15)	No	binding

Purchases

	Column	Type	Null	Default
Pk	***id***	int(11)	No	
Fk	book_id	int(11)	No	
	quantity	int(11)	No	
	price	int(11)	No	
	p_date	date	No	

Scanned with CamScanner

מסמך5



Scanned with CamScanner

ממן 6

Books

id	title	Auther_name	Translator_name	Release_name	Release_year	Pages_number	weight	condition	price	Quantity_in_store	Created_at
1	book1	auther3		book1	2020	19	200	new	30	5	2020-07-06
2	book1	auther4 ,aurther5		book2	2020	19	200	old	15	4	2020-07-05
3	book2	author1, author2		book2	1995	50	350	new	45	0	2020-07-06
4	book2	author1 ,author2	translator1	book2 arabic	1995	354	350	good	60	0	2020-07-03

Costumers

id	fname	lname	address	Phone_number	Created_at
123	customer1	customer1	address1	123456789	2020-07-01
456	customer2	customer2	address2	234567891	2020-07-02
789	customer3	customer3	address3	345678912	2020-07-03

Deals

id	S_date	Costumer_id	Purchase_id	Employee_id
1	2020-07-09	123	credit card	1111
2	2016-07-16	456	bit	2222
3	2020-06-21	789	bank bass	2222
4	2020-07-08	789	Bit	2222

Scanned with CamScanner

deals_deatails

id	Book_id	Deal_id	price
1	1	1	30
2	2	1	15
3	3	2	45
4	4	3	60
5	3	4	14
6	4	4	60

deliveries

id	S_date	address	type	price	status
1	2020-07-21	address1	post	16	binding
2	2020-07-21	address2	xpress	46	binding
3	2020-07-09	address3	post express	49	binding
4	2020-07-07		xpress	30	binding

Deals_details_deliveries

id	Deal_details_id	Delivery_id
1	1	1
2	2	2
3	3	3
4	5	4
5	6	4

employees

id	fname	lname	address	Phone_number	S_date	E_date	Hour _price
1111	employee1	employee1	address1	1122334455	2020-07-08	NULL	29
2222	employee2	employee2	address2	2233445566	2020-07-06	NULL	34
3333	employee3	employee3	address3	3344556677	2020-07-03	2020-07-22	31

Scanned with CamScanner

Working_hours

id	Worker_id	W_year	W_month	quantity
1	1111	2020	6	135
2	1111	2020	7	122

Orders

id	Book_id	Costumer_id	Calling_date	Created_at	status
1	1	123	NULL	2020-07-25	binding
2	4	789	2020-07-04	2020-07-01	binding

purchases

id	Book_id	quantity	price	P_date
1	1	20	10	2020-07-01
2	2	15	9	2020-07-01
3	3	12	11	2020-06-01
4	4	16	8	2020-06-02

Scanned with CamScanner

מסמך 6.1

1. SELECT * FROM books WHERE title = 'auther' OR auther_name like '%auther%'

2. SELECT * FROM `customers` ORDER BY created_at LIMIT 1

3. SELECT * FROM `books` ORDER BY created_at LIMIT 1

4. SELECT * FROM orders WHERE status = 'binding' ORDER BY created_at

5. SELECT books.id, books.title, books.auther_name, books.translator_name, books.release_name, books.release_year, books.condition, COUNT(deals_details.id) AS 'counts'
   FROM deals_details
   JOIN books ON deals_details.book_id = books.id
   WHERE books.title = 'auther' OR books.auther_name like '%auther%'
   GROUP BY books.id, books.title, books.auther_name, books.translator_name, books.release_name, books.release_year, books.condition

6. SELECT books.auther_name,COUNT(books.id)
   FROM deals_details JOIN books ON deals_details.book_id = books.id

Scanned with CamScanner

```
JOIN deals ON deals_details.deal_id = deals.id
WHERE deals.s_date BETWEEN '2020-01-01' AND '2020-01-01'
GROUP BY books.auther_name ORDER BY COUNT(books.id)
```

7.
```
SELECT customers.*, COUNT(deals_details.id) as 'reads'
FROM deals_details
JOIN deals ON deals_details.deal_id = deals.id
 JOIN customers ON deals.customer_id = customers.id
GROUP BY customers.id, customers.fname, customers.lname,
customers.address, customers.phone_number,
customers.created_at
ORDER BY COUNT(deals_details.id) DESC
LIMIT 3
```

8.
```
SELECT books_translated.title, COUNT(books_translated.title)
FROM(
    SELECT books.title, books.translator_name
    FROM books
    GROUP BY books.title, books.translator_name) AS
books_translated
GROUP BY books_translated.title
ORDER BY COUNT(books_translated.title) DESC
LIMIT 1
```

Scanned with CamScanner

9. SELECT books.*, deals.s_date

   FROM deals_details

   JOIN deals ON deals_details.deal_id = deals.id

   JOIN books ON deals_details.book_id = books.id

   WHERE cast(deals.customer_id AS CHAR) = '123'

   ORDER BY deals.s_ date

10. SELECT books.id,books.title,books.translator_name,

    orders.created_at,

    IF(books.id IN(SELECT purchase.book_id FROM purchase), 'yes',

    'no') AS 'in_inventory', IF(books.id IN(SELECT

    deals_details.book_id

                FROM deals_details

                JOIN deals ON deals_details.deal_id = deals.id

                WHERE deals.customer_id = 123), 'yes', 'no') AS

    'deal'

    FROM books

    JOIN orders ON orders.book_id = books.id

    WHERE orders.customer_id = 123

11. ש

12. SELECT customers.id, CONCAT(customers.fname,'

    ',customers.lname) AS name,deliveries.*

    FROM customers

    JOIN deals ON deals.customer_id = customers.id

    JOIN deals_details ON deals_details.deal_id = deals.id

    JOIN deliveries_details ON deliveries_details.deal_details_id =

    deals_details.id

Scanned with CamScanner

```
JOIN deliveries ON deliveries_details.delivery_id = deliveries.id
WHERE deals.id IN(
        SELECT deals.id
         FROM customers
         JOIN deals ON deals.customer_id = customers.id
         JOIN deals  details ON deals  details.deal  id = deals.id
              JOIN deliveries_details ON deliveries_details.deal_details_id
    = deals_details.id
         GROUP BY customers.id, CONCAT(customers.fname, ' ',
    customers.lname),         deals.id
         HAVING COUNT(deliveries_details.id) >= 2)
```

13.
```
SELECT deliveries.*
    FROM deals
    JOIN deals_details ON deals_details.deal_id = deals.id
    JOIN deliveries_details ON deliveries_details.deal_details_id =
    deals_details.id
    JOIN deliveries ON deliveries_details.delivery_id = deliveries.id
    WHERE deals.customer_id = 123
```

14.
```
SELECT COALESCE(SUM(deliveries.price),0)
    FROM deliveries WHERE deliveries.type = 'xpress'
    AND MONTH(deliveries.s_date) = 8
```

Scanned with CamScanner

15.SELECT COALESCE(SUM(deals_details.price),0)

FROM deals JOIN deals_details ON deals_details.deal_id = deals.id

WHERE deals.purchase_type = 'Bit'

AND MONTH(deals.s_date) = 77

16.SELECT deals.id, SUM(deals_details.price) as avrage

FROM deals JOIN deals_details ON deals_details.deal_id = deals.id

WHERE deals.s_date >= DATE_SUB(NOW(), INTERVAL 1 YEAR)

GROUP BY deals.id

HAVING average > (

SELECT AVG(sum_deals.avrage)

FROM(

SELECT deals.id, SUM(deals_details.price) as average

FROM deals JOIN deals_details ON

deals_details.deal_id = deals.id

WHERE deals.s_date >= DATE_SUB(NOW(), INTERVAL 1

YEAR)

GROUP BY deals.id) AS sum_deals)

17.SELECT deliveries.type, COUNT(deliveries.id)

FROM deliveries

WHERE deliveries.s_date >= DATE_SUB(NOW(), INTERVAL 1 YEAR)

AND(deliveries.type = 'xpress' OR deliveries.type = 'post')

GROUP BY deliveries.type

Scanned with CamScanner

18. SELECT deliveries_details.delivery_id, books.title,
    COUNT(books.id) AS releases_number FROM deliveries_details
    JOIN deals_details ON deals_details.id =
    deliveries_details.delivery_id
    JOIN books ON books.id = deals_details.book_id
    GROUP BY deliveries_details.delivery_id, books.title,
    books.release_name
    HAVING COUNT(books.id) >= 2

19. SELECT DISTINCT customers.*
    FROM customers
    JOIN deals ON deals.customer_id = customers.id
    WHERE customers.id AND deals.s_date < DATE_SUB(NOW(),
    INTERVAL 2 YEAR)

20. SELECT DISTINCT customers.*
    FROM customers
    JOIN orders ON orders.customer_id = customers.id
    WHERE orders.calling_date < DATE_SUB(NOW(), INTERVAL 2
    WEEK)

21. SELECT DISTINCT YEAR(purchase.p_date) AS a_year,
    MONTH(purchase.p_date) AS a_month, (
        SELECT SUM(purchase.quantity)
        FROM purchase
        WHERE YEAR(purchase.p_date) <= a_year

Scanned with CamScanner

```
                AND MONTH(purchase.p_date) <= a_month)  -
               (SELECT COUNT(deals_details.id)
               FROM deals_details  JOIN deals ON
deals_details.deal_id = deals.id
                WHERE YEAR(deals.s_date) <= a_year
                AND MONTH(deals.s_date) <= a_month) AS
    books_count
    FROM purchase
```

22. SELECT SUM(purchase.quantity),
    SUM(purchase.quantity*purchase.price)
    FROM purchase
    WHERE purchase.p_date BETWEEN '1-1-2020' AND '30-9-2020'

23. SELECT COALESCE(
        (SELECT SUM(purchase.price * purchase.quantity)
        FROM purchase
        WHERE MONTH(purchase.p_date) = 7
        AND YEAR(purchase.p_date) = 2020) –
        (SELECT SUM(deals_details.price)
        FROM deals_details
        JOIN deals ON deals.id = deals_details.deal_id
        WHERE MONTH(deals.s_date) = 7
        AND YEAR(deals.s_date) = 2020),0) AS store_pro

Scanned with CamScanner

24.SELECT monthly_sum.a_year, AVG(monthly_sum.a_sum)

FROM (

SELECT MONTH(deals.s_date) AS a_month,

YEAR(deals.s_date) AS a_year,

SUM(deals_details.price) AS a_sum

FROM deals

JOIN deals_details ON deals.id = deals_details.deal_id

GROUP BY MONTH(deals.s_date), YEAR(deals.s_date)) AS

monthly_sum

GROUP BY monthly_sum.a_year

25.SELECT COALESCE(SUM(working_hours.quantity *

employees.hour_price),0) AS salary FROM employees JOIN

working_hours ON working_hours.worker_id = employees.id

WHERE working_hours.w_month = 7

AND working_hours.w_year = 2020 AND employees.id = '123'

26.SELECT employees.id, employees.fname, employees.lname

FROM employees JOIN deals ON deals.employee_id =

employees.id

WHERE MONTH(deals.s_date) = 8 AND YEAR(dealss_date) = 8

ORDER BY COUNT(deals.id) DESC

LIMIT 1

Scanned with CamScanner

מסמך 6.1

(1)

Result ← σ book

"title='x' or author_name="x

(2)

Minemal(minimal_date) ← F min (customers)

created_at

Result ← σ (customers ⋈ minimal)

created_at= minimal_date

(3)

Minimal(minimal_date) ← F min (books)

created_at

Result ← σ (books ⋈ minimal)

created_at=minimal_date

(4)

T1 ← τ orders

created_at

Result ← σ (orders ⋈ customers)

status="binding" customer_id

Scanned with CamScanner

(5)

T1 ← books ⋈ purchase

book_id

T2 ← σ T1

"title='y' or author_name="y

Result(counter) ← F count (T2)

book_id

(6)

T1 ← σ orders

x ≤ calling_date ≤ y

T2 ← books ⋈ T1

book_id

T3(author_name, count) ← F count (T2)

author_name

T4(max) ← F max (T3)

counter

Result ← σ T3

count= max

Scanned with CamScanner

(7)

T1 ← customers ⋈ deals

customer_id

T2(customer_id, counter) ← Z count (T1 ⋈ deals_details)

book_id deal_id

T3(customer_id, max0) ← Z max (T2)

counter

T4(customer_id, max1) ← Z max (T2 - T3)

counter

T5(customer_id, max2) ← Z max (T2 - T3 - T4)

counter

Result ← [(T3 ⋈ T4 ⋈ T5) ⋈ customers]

customer_id customer_id customer_id

(8)

T1(book_id, counter) ← F count (books)

translator_name

T2(max) ← F max (T1)

counter

T3 ← σ (T1)

max= counter

Result ← σ (book ⋈ T3)

book_id

Scanned with CamScanner

(9)

T1 ← σ customers

customer_name='x'

T2 ← deals ⋈ T1

customer_id

T3 ← deals_details ⋈ deals

deals_id

Result ← T1 (τ T3)

customer_id,book_id,title,s_date,price s_date

(10)

T1 ← τ orders

created_at

T2 ← σ (customer ⋈ T1)

customer_name='x' customer_id

T3 ← T2 ⋈ purchase

book_id

Result ← T3 ⋈ deals_details

book_id

Scanned with CamScanner

(12)

T1 ← deliveries_details ⋈ deals_details

deal_details_id

T2 ← T1 ⋈ deals

deal_id

T3 ← T2 ⋈ deliveries

delivery_id

Result ← σ T3

customer_id='x'

(14)

T1(deliveries.*, sum) ← F sum deliveries

price

Result ← σ T1

type='xpress' and s_date='8

(15)

T1 ← deals ⋈ deals_details

deal_id

T2(T1.*, sum) ← F sum (T1)

price

Result ← σ T2

purchase_type="bit" and s_date

Scanned with CamScanner

(17)

T1 (deliveries.*, count) ← F count_id(deliveries)

Result ← π (σ T1)

type,count s_date≥date- 1year and (type='xpress' or type='post')

(18)

T1 ← deliveries_details ⋈ deals_details ⋈ books

delivery_id book_id

T2(T1.*, count) ← F count T1

book_id

Result ← T1 (σ T2)

delivery_id,title,count count≥2

(19)

T1 ← customers ⋈ deals

customer_id

Result ← π (σ T1)

customer.* s_date<date-sub(nowl)interial.2year

(20)

T1 ← customers ⋈ orders

customer_id

Result ← π (σ T1)

customer.* calling_date<date-sub(nowl)interial. 2 week

Scanned with CamScanner