Разработать процедуру, которая будет принимать на вход дату. В процедуре для всех заказов этого дня реализовать занесение соответствующих строк в таблицу Расписание аренды.

|  |  |  |
| --- | --- | --- |
| Bilbord - билборд | | |
| id\_bil | Address | price |
| PK |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Timetable - расписание | | | | | |
| id\_table | id\_bil | s\_y | s\_m | f\_y | f\_m |
| PK | FK |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Order - заказы | | | | | |  |  |
| id\_order | id\_bil | dat\_z | s\_y | s\_m | f\_y | f\_m | all\_price |
| PK | FK |  |  |  |  |  |  |

План процедуры:

1. Декларируем локальные переменные id\_bilb, year\_s, month\_s, year\_f, month\_f;
2. Декларируем курсор;

DECLARE C1 CURSOR FOR

SELECT id\_bil, s\_y, s\_m, f\_y, f\_m FROM order1

WHERE dat\_z = userdate;

1. Декларируем обработчик для отслеживания конца курсора;

DECLARE exit handler for SQLSTATE '02000' set done = 1;

1. Открываем курсор;
2. Открываем цикл чтения записей из курсора;

fetch C1 into id\_bilb, year\_s, month\_s, year\_f, month\_f;

1. Каждая запись о заказе, подходящая по дате, переносится в расписание аренды;

INSERT timetable values(NULL, id\_bilb, year\_s, month\_s, year\_f, month\_f);

1. По достижению конца курсора производится выход из цикла и последующее закрытие курсора.

CREATE PROCEDURE `z`(userdate date)

BEGIN

DECLARE id\_bilb, year\_s, month\_s, year\_f, month\_f integer;

DECLARE done integer default 0;

DECLARE C1 CURSOR FOR

SELECT id\_bil, s\_y, s\_m, f\_y, f\_m FROM order1

WHERE dat\_z = userdate;

DECLARE exit handler for SQLSTATE '02000' set done = 1;

OPEN C1;

while done = 0 do

fetch C1 into id\_bilb, year\_s, month\_s, year\_f, month\_f;

INSERT timetable values(NULL, id\_bilb, year\_s, month\_s, year\_f, month\_f);

END while;

Close C1;

END

Разработать триггер, который будет срабатывать на занесение новой записи в таблицу Заказы. В триггере реализовать занесение соответствующей записи в таблицу Расписание аренды

CREATE DEFINER = CURRENT\_USER TRIGGER `rk`.`order1\_AFTER\_INSERT` AFTER INSERT ON `order1` FOR EACH ROW

BEGIN

DECLARE id\_bilb, year\_s, month\_s, year\_f, month\_f integer;

DECLARE done integer default 0;

DECLARE C1 CURSOR FOR

SELECT id\_bil, s\_y, s\_m, f\_y, f\_m FROM order1

WHERE id\_bil = new.id\_bil;

DECLARE exit handler for SQLSTATE '02000' set done = 1;

OPEN C1;

while done = 0 do

fetch C1 into id\_bilb, year\_s, month\_s, year\_f, month\_f;

INSERT timetable values(NULL, id\_bilb, year\_s, month\_s, year\_f, month\_f);

END while;

Close C1;

END

Разработать процедуру, которая будет принимать на вход дату, выбирать из таблицы Поставки все записи за указанный день и обновлять количество товара и дату обновления для соответствующих записей в таблице Наличие товара.

|  |  |  |
| --- | --- | --- |
| Product - товар | | |
| id\_product | Title | UM |
| PK |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Available – наличие товара | | | |
| id\_avialable | id\_product | count\_product | date\_update |
| PK | FK |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Deliveries - поставки | | | | |
| id\_deliveries | id\_product | id\_sup | count\_product | date\_deliveries |
| PK | FK |  |  |  |

План процедуры:

1. Декларируем локальные переменные coun, id\_p integer;
2. Декларируем курсор;

DECLARE C1 CURSOR FOR

SELECT count\_product, id\_product FROM deliveries

WHERE date\_deliveries = userdate;

1. Декларируем обработчик для отслеживания конца курсора;

DECLARE exit handler for SQLSTATE '02000' set done = 1;

1. Открываем курсор;
2. Открываем цикл чтения записей из курсора;

fetch C1 into coun, id\_p;

1. Каждая запись о заказе, подходящая по дате, обновляется в наличии товара;

UPDATE available set count\_product=count\_product+coun, date\_update=userdate where id\_product = id\_p;

1. По достижению конца курсора производится выход из цикла и последующее закрытие курсора.

CREATE PROCEDURE `z`(userdate date)

BEGIN

DECLARE coun, id\_p integer;

DECLARE done integer default 0;

DECLARE C1 CURSOR FOR

SELECT count\_product, id\_product FROM deliveries

WHERE date\_deliveries = userdate;

DECLARE exit handler for SQLSTATE '02000' set done = 1;

OPEN C1;

while done = 0 do

fetch C1 into coun, id\_p;

UPDATE available set count\_product=count\_product+coun, date\_update=userdate where id\_product = id\_p;

END while;

Close C1;

END

Разработать триггер, который будет срабатывать на добавление записи в таблицу Поставки. В триггере реализовать обновление количества товара и дату обновления для соответствующей записи в таблице Наличие товара.

CREATE TRIGGER `deliveries\_AFTER\_INSERT` AFTER INSERT ON `deliveries` FOR EACH ROW BEGIN

DECLARE coun, id\_p integer;

DECLARE dat date;

DECLARE done integer default 0;

DECLARE C1 CURSOR FOR

SELECT count\_product, id\_product, date\_deliveries FROM deliveries

WHERE id\_product = new.id\_product;

DECLARE exit handler for SQLSTATE '02000' set done = 1;

OPEN C1;

while done = 0 do

fetch C1 into coun, id\_p, dat;

UPDATE available set count\_product=count\_product+coun, date\_update = dat where id\_product = id\_p;

END while;

Close C1;

END

Разработать процедуру, которая получает на вход дату, выбирает все поставки книг за указанный день и обновляет количество соответствующих книг и дату в таблице Запас книг.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Deliveries - поставки | | | | |
| id\_deliveries | id\_public | id\_book | count\_book | date\_deliveries |
| PK | FK | FK |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| public – наличие товара | | | |
| id\_public | title\_p | date\_zakl | year |
| PK | FK |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Book – запас книг | | | | |
| id\_book | avtor | title\_b | count\_book | date\_update |
| PK |  |  |  |  |

План процедуры:

1. Декларируем локальные переменные coun, id\_b integer;
2. Декларируем курсор;

DECLARE C1 CURSOR FOR

SELECT count\_book, id\_book FROM deliveries

WHERE date\_deliveries = userdate;

1. Декларируем обработчик для отслеживания конца курсора;

DECLARE exit handler for SQLSTATE '02000' set done = 1;

1. Открываем курсор;
2. Открываем цикл чтения записей из курсора;

fetch C1 into coun, id\_b;

1. Каждая запись о заказе, подходящая по дате, обновляется в наличии товара;

UPDATE book set count\_book=count\_book+coun, date\_update=userdate where id\_book = id\_b;

1. По достижению конца курсора производится выход из цикла и последующее закрытие курсора.

CREATE PROCEDURE `z`(userdate date)

BEGIN

DECLARE coun, id\_b integer;

DECLARE done integer default 0;

DECLARE C1 CURSOR FOR

SELECT count\_book, id\_book FROM deliveries

WHERE date\_deliveries = userdate;

DECLARE exit handler for SQLSTATE '02000' set done = 1;

OPEN C1;

while done = 0 do

fetch C1 into coun, id\_b;

UPDATE book set count\_book=count\_book+coun, date\_update=userdate where id\_book = id\_b;

END while;

Close C1;

END

Разработать триггер, который при появлении новой записи в таблице Поставки будет обновлять существующую или добавлять новую запись в таблицу Запас книг.

CREATE TRIGGER `deliveries\_AFTER\_INSERT` AFTER INSERT ON `deliveries` FOR EACH ROW BEGIN

DECLARE id\_b integer;

DECLARE done integer default 0;

DECLARE C1 CURSOR FOR

SELECT id\_book FROM book

WHERE id\_book = new.id\_book;

DECLARE exit handler for SQLSTATE '02000' set done = 1;

OPEN C1;

while done = 0 do

fetch C1 into id\_b;

if id\_b is NULL THEN

INSERT book VALUES(new.id\_book, NULL,NULL,new.count\_book, new.date\_deliveries);

ELSE

UPDATE book set count\_book=count\_book+new.count\_book, date\_update=new.date\_deliveries where id\_book = id\_b;

END if;

END while;

Close C1;

END