WEB APPLICATION AND SERVICE
CAND NO. - 297285

Question 1

i) True

The client also receives an acknowledge when TCP connection is made but that is not in the application layer. HTTP/1.0 and HTTP/1.1 will receive 5 responses to 5 GET message. Whereas HTTP/2.0 uses streams to send chunks of data so might receive more than 5 responses if file size requested is large enough.

ii) True

Provided the connection remain open between the server and client and page are hosted over the same server (www.sussex.ac.uk). HTTP/1.1 can be used for persistent connections or 'Connection: Keep-Alive' header in case of HTTP/1.0

iii) True

'Accept' header in the request describe the format of content the client wants to receive from the server. It is used for content negotiation. ~~text~~ Certain request responses are required in a specific format to understood by the request client (eg application/json, application/xml).

iv) False

Responses might have empty body if proper messages are not configured to be displayed. 500 Internal server Error or 204 No Content the responses with these status codes usually have an empty body. HEAD request also have no body.

1.b) The following is REST service is being used to update a guest resource.

REQUEST:

→ PUT Hotel/guest/..... (1 line)

The PUT HTTP method is used to update an existing resource.

Hotel/guest/bc45-9aa3-3f22 is the path to resource of guest-id (bc45.....) which is an unique identifier.

HTTP 1.1 - The version of HTTP protocol used

→ Host: www.somehotelservice.com - denotes the domain name of server to which a connection (TCP) is made and request is made

→ Content-Length - 94 : maximum size of message body can be 94 bytes

→ Content-Type - This denotes the format in which the request body is coming in as

application/x-www-form-urlencoded which is body contains key-value pairs separated by '&'.

→ zip=30314&lastName=Doe..... - The message body containing the field to be modified for the user-id "bc45-9aa3-3f2d".

Response :

→ HTTP/1.1 200 OK

↓ HTTP protocal used

↓ status code and phrase indicating request was successful

→ Content-Length :36 (header) response message body size is 36 bytes

→ Content-Type : text/plain -(header) specifying the media type of the response body.

→ Location : http://..... - confirms the URI of the resource acted upon.

→ The guest resource has been updated - The message response body in plain text format.

1.c) Django Models defined here are:

→ Customer

has two fields

- ssnumber - which is a primary key (non-null) with a maximum length of 10 characters. charfield signifies it can be a combination of alphanumeric character and some symbols too. Primary keys uniquely identify a customer.
- name - It is also a character field with maximum length 100 characters.

→ Order has 2 fields

- customer - Foreign Key which connects the Order and the Customer model. This depicts a Many-to-One relationship signifying many orders can belong to a customer. The customer column stores the primary key of the customer (ssnumber) with which it can be queried. on_delete = models.CASCADE means that if a customer is deleted then all orders associated with it are deleted.
- address - stores the address where order needs to be delivered which is 100 letter sequence of characters

Example :

Customer Table

ss number (PK)	name
123	KB
456	GTR

Order

id (PK)	customer (FK)	address
1	123	123 Main
2	456	234 Cross
3	123	789 Pine

PK - Primary key FK - foreign key

The tables are created based on the models defined in forms.py.

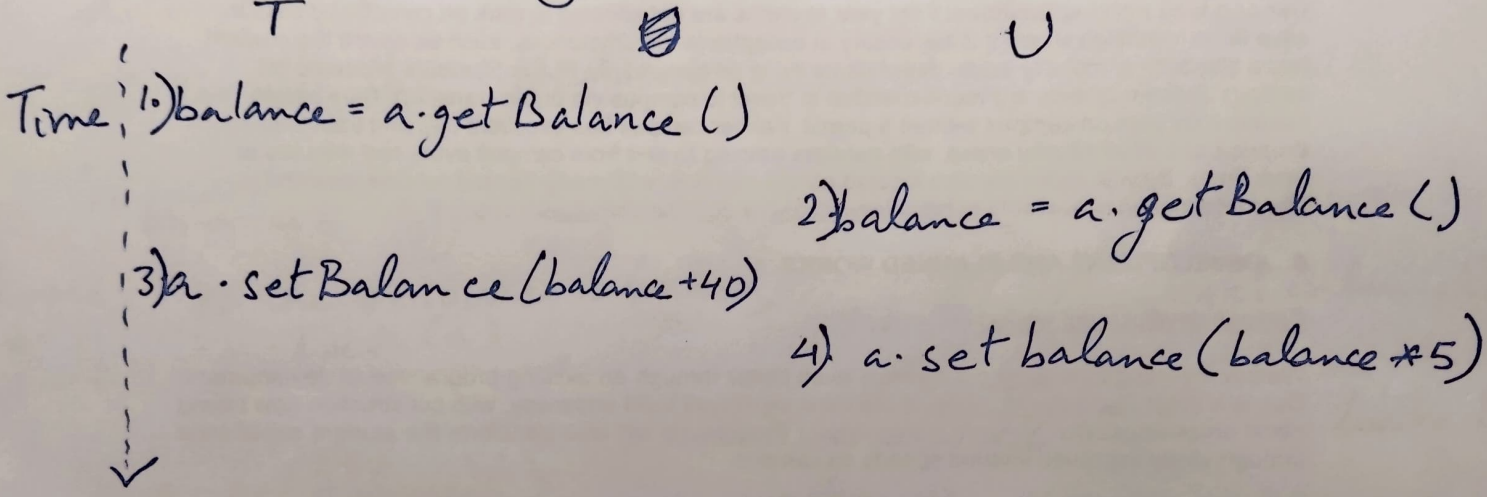
The Order defines a primary key to maintain uniqueness of each row.

The customer (FK) field hold the ssnumber of the customer table and also shows a Many-to-one relationship between order and customer (1 customer may have many orders)

On deleting 123 customer it'll delete order id 1 and 3.

Ques 3

a) i) Lost update happens when both transactions read the same data and update the data based on the read but one of the updates get ~~overriden~~ overwritten by the other.



For the sequence the outputs will be:
let's say ~~a.get~~ a.balance = 100

1. read - 100 → T
2. read - 100 → U
3. update - 140 → T (100+40) {This update is lost}
4. update - 100*5 = 500 → U

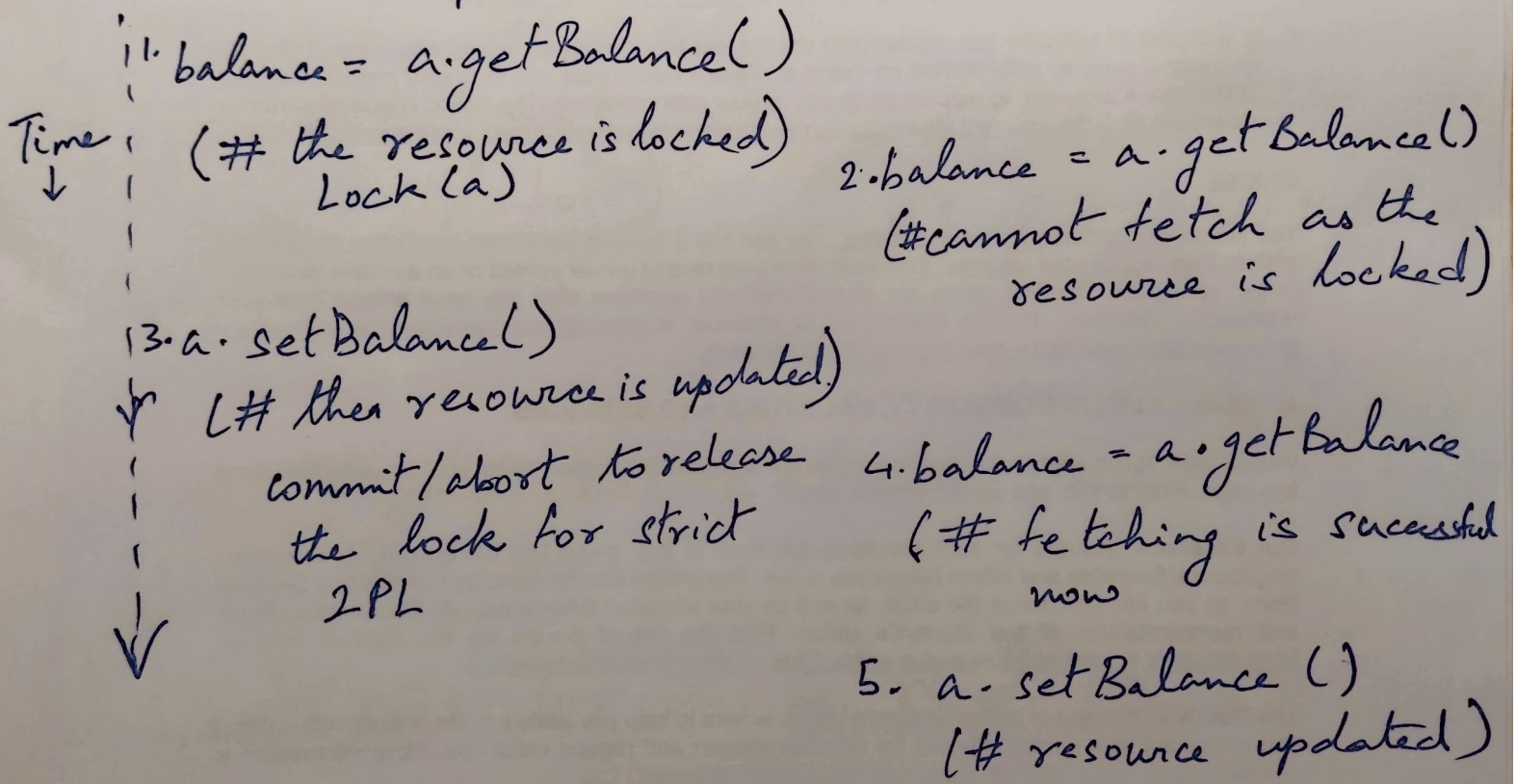
Prevention using 2PL & exclusive locks:
Two phase ensures total serializability while exclusive locks ensure one transaction can hold a lock on a data item at a time of writing.

a) Growing Phase - Transaction acquire locks during read.

b) Shrinking Phase - Lock is released after write. For strict 2PL the lock is released only after a commit or abort

Locking mechanism

a. balance = 100 is the shared resource so it will be locked

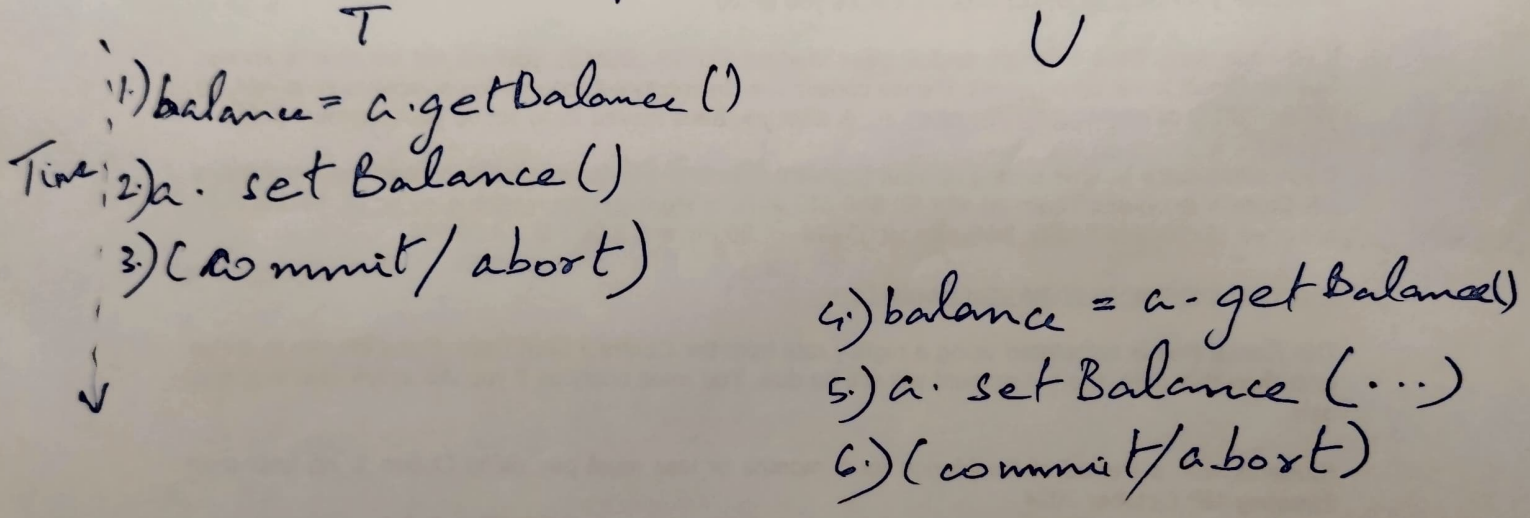


ii) The dirty Read cannot happen if the 2 transactions are executed in a serially equivalent manner. (one-by-one)

Dirty read occurs when a transaction reads data that has been written by another transaction that has not

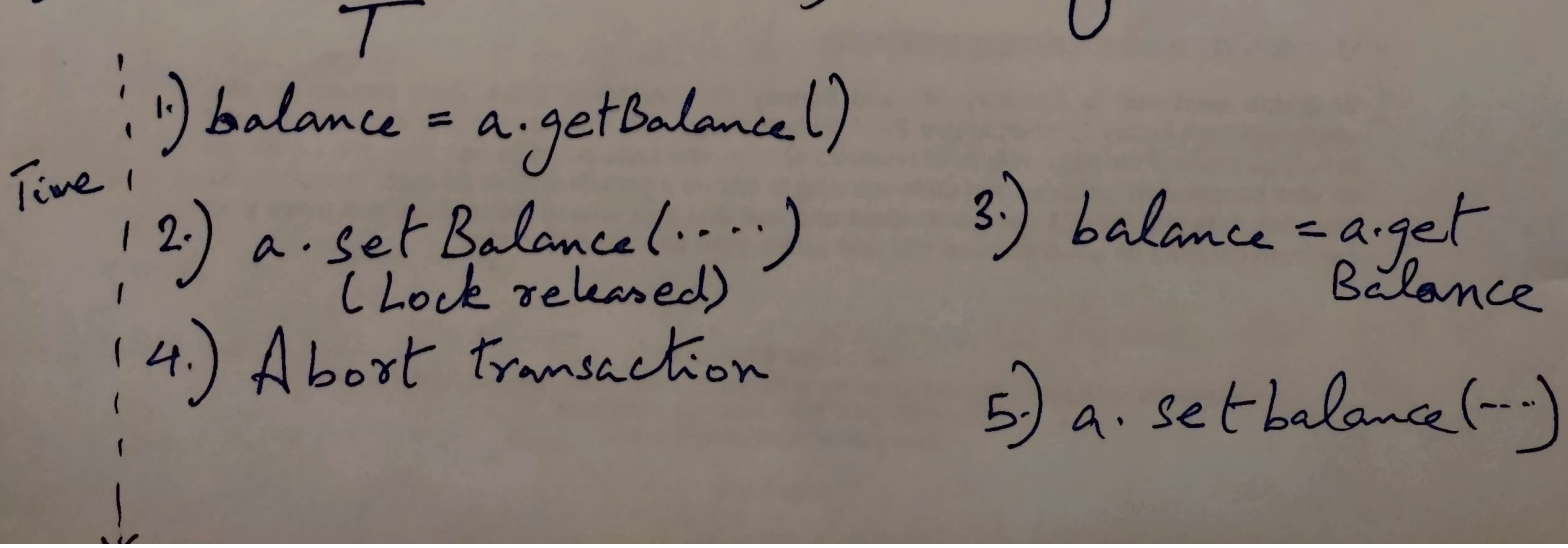
committed or if a writing transaction is aborted but the reading transaction uses the uncommitted data leading to inconsistency.

Since T is independently executed and it's results are committed or aborted properly, U will pick up the correct value



If read happens before a (commit/abort) it might cause a dirty Read.

iii) Scenarios for dirty reads



In this case the updates from transaction T are rolled back but the transaction U reads the updated value from transaction T leading to a dirty read

Strict 2PL the lock is acquired during a read but released only after a commit or abort is done. X Lock (exclusive lock) is used by T until a commit/abort happen. U can read data using S Lock but the changes are rolled back completely in case of update.

3.b) i) Differences -

→ transmission Time - Synchronous systems have a defined timeout whereas async systems don't

→ Implication of failed process - For a synchronous system if a reply doesn't come it mean a failure has occured. For async system the response time might be slow or a crash might happen this is not distinguishable.

ii) Election Process (24 → 46 → 5 → 88 → 32 → 24)

1) P24 initiates Election becomes a participant Message ELECTION(24). sends to 46

2) P46 compares id 46 > 24. P46 become participant ELECTION(46) message sent to P5

3) P5 check id 5 < 46. P5 forwards message to P88 ELECTION(46) and becomes participant

4) P88 checks id 88 > 46. So it sends message ELECTION(88) to 32 and becomes participant

This continues until the message reaches P88 again where message id is equal to process id. Thus P88 becomes the Leader transitions to non-participant and send COORIDINATOR(88) message around the ring to transition all participant to non participants.

c) For status updates on a social network appearing on user's walls a combination of of ordering is desirable with Casual Ordering being most important followed by FIFO ordering. Casual ordering ensures comments and related posts makes sense where as FIFO for posts from same individual or threads of replies maintain narrative flow.