**Question 1**

A candidate key is a minimal set of columns which uniquely identify rows in a table. A primary key is an arbitratily selected candidate key. Proper subsets of candidate keys are subsets of a CK's attributes which are proper subsets i.e. have cardinality strictly smaller than the original set.

1 mark for each

**Question 2**

a) Employee age is stored in two places. On the employee's birthday, we try to run two UPDATE statements to add one to their age. The database connection fails and the second statement is not run, leaving the database in an inconsistent state.

b) Normalisation (2NF or 3NF) mandates that the employee age is stored only in one table, so it can be updated atomically.

1 mark for each

**Question 3**

a) 1NF mandates no lists of records in table cells.

b) No, there is still only one address record present.

1 mark for each

**Question 4**

a) Snake and camel case not consistent.

b) Name should be TEXT, ID should be INTEGER or SERIAL, room should be a string, number or foreign key (INTEGER or TEXT), duration should be time (date has no time info), number is CORRECT as string to capture leading zeros, country code etc.

1 mark for names
1 mark each for datatypes
deduct one if the student corrected the phone number (which is correct as string) but it's also correct for them to correct it to TEXT

**Question 5**

Problems include but are not limited to data loss, data sharing, data corruption, security, ease of analysis and querying, speed, and schema normalisation. 2 marks per problem for clear statement and resolution.

Question 6

INNER JOIN only shows rows where there is a match and is symmetric. LEFT OUTER JOIN shows rows from both tables and is not symmetric.

1 for each and 1 for symmetricity.

**Part B**

**Question 1**

Yes, members, blocks and opinions.
1 mark for any correct answer, 1 for justification.

**Question 2**

No, this relationship is not symmetric, because both records could exist separately.
1 mark for correct answer; 1 for justification.

**Question 3**

```
SELECT users.*, COUNT(*) AS msg_count
FROM
users INNER JOIN messages
ON users.id = messages.user_id
GROUP BY users.id
```

**Question 4**

```
SELECT chats.*, COUNT(*) as message_count FROM
chats INNER JOIN messages
ON chats.id = messages.chat_id
GROUP BY chats.id
```

**Question 5**

```
SELECT users.id, MAX(messages.created_at)
FROM
users INNER JOIN messages
GROUP BY users.id
```

**Question 6**

```
WITH last_messages AS

SELECT users.id AS user_id, MAX(messages.created_at)
```

```
FROM
users INNER JOIN messages
GROUP BY users.id

SELECT users.id, users.name, messages.created_at, messages.text FROM
users INNER JOIN last_messages
ON users.id = last_messages.user_id
AND
last_messages.created_at = messages.created_at
```

**Question 7**

```
a) SELECT * FROM
users INNER JOIN
(SELECT id, COUNT(*) AS msg_count
users INNER JOIN messages
ON users.id = messages.user_id
GROUP BY users.id
ORDER BY msg_count
LIMIT 1) as top_msg
```

This could also be done in other ways (CTEs) but must be done in one query (no views)

b)

This depends how it is done, but in the query above, we would get one of the users nondeterministically. If done otherwise, there may be two rows shown.

2 marks per part, 1 for answer and 1 for query.

**Question 8**

```
SELECT messages.*, LENGTH(text) as msg_length,
SUM(LENGTH(text)) OVER (ORDER BY created_at) AS msg_sum
FROM messages
WHERE user_id = 777
ORDER BY created_at
```

1 mark for length col
1 mark for window function
1 mark for WHERE
1 mark for ORDER BY in WF
1 mark for ORDER BY at end

**Question 9**

```
SELECT COUNT(DISTINCT block_ID) AS block_count,
COUNT(DISTINCT opinions.id) as neg_opinion_count
FROM
users INNER JOIN user_blocks
ON users.id = user_blocks.user_id
INNER JOIN opinions
ON users.id = opinions.user_id
WHERE opinions.opinion IS FALSE

GROUP BY users.id
```

2 marks for joins
1 mark for DISTINCT
1 mark for WHERE

**Question 10**

a) No, it is not recorded who blocked whom.

b) Yes:

```
SELECT * FROM opinions
WHERE target_id = 777
AND opinion = FALSE
```

2 marks for a (1 mark for not giving a query)
2 marks for b (one for query)