1. We apply the lossless join test. The tableau corresponding to the decomposition $\rho$ is:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>b</td>
<td>c</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>c</td>
<td>d</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
</tr>
</tbody>
</table>

After chasing this with respect to $F = \{B \rightarrow A, C \rightarrow B\}$ the last row becomes $<a, b, c, d>$. 

2. The fds $AB \rightarrow C, C \rightarrow E, E \rightarrow C$ are obviously preserved because each applies to one relation in the decomposition. Consider $C \rightarrow D$, which does not apply to a single relation. We compute the closure of $C$ relative to the local fds according to the algorithm described in class. Initially we have $C$ in the relation $CE$ and in $ABC$. The closure of $C$ is $CED$, so we obtain $E$ within $CE$. Now $E$ is available in $ADE$. The closure of $E$ is $ECD$ so we obtain $D$ within $ADE$. Since $D$ is now on the list, it is in the closure of $C$ wrt the local fds, so $C \rightarrow D$ is preserved. It remains to check $AB \rightarrow E$. We now start with $AB$, so we have $AB$ within $ABC$, and $A$ within $ADE$. The closure of $AB$ is $ABCD$ so we obtain $C$ within $ABC$. Now we have $C$ within $CE$, and $C^+ = CED$ so we get $E$ within $CE$. Thus, $AB \rightarrow E$ is also preserved.

3. We first rewrite the fds so that we only have single attributes on the righthand side:

$$A \rightarrow C, AB \rightarrow C, C \rightarrow I, C \rightarrow D, CD \rightarrow I, EC \rightarrow A, EC \rightarrow B, EI \rightarrow C$$

We next look at each of the fds and see if they are redundant. $A \rightarrow C$ is not, because $A^+$ (wrt the other fds) is $A$. $AB \rightarrow C$ is clearly redundant, since it is implied by $A \rightarrow C$. We eliminate it from the list. Similarly, $C \rightarrow D$ is not redundant. However, $C \rightarrow I$ is redundant, and we eliminate it. Next, $CD \rightarrow I$ is not redundant wrt the fds left on the list. Similarly, $EC \rightarrow A$, $EC \rightarrow B$, and $EI \rightarrow C$ are not redundant. The remaining list of fds is so far:

$$A \rightarrow C, C \rightarrow D, CD \rightarrow I, EC \rightarrow A, EC \rightarrow B, EI \rightarrow C.$$
Next, we check for redundant attributes on left-hand sides of fds. Consider $CD \rightarrow I$. We need to check whether $C$ or $D$ can be eliminated. $C$ can be eliminated if $D \rightarrow I$ is implied by the fds on the list (the entire list!). Clearly, $D^+ = D$, so $D \rightarrow I$ is not implied. Next, $D$ can be eliminated if $C \rightarrow I$ is implied. Now $C^+ = CDI$ so $C \rightarrow I$ is implied. So $D$ is redundant and we replace $CD \rightarrow I$ by $C \rightarrow I$ in the list of fds. It easy to see that there are no redundant attributes in

$$EC \rightarrow A, EC \rightarrow B, EI \rightarrow C$$

so the final minimized set of fds is:

$$A \rightarrow C, C \rightarrow D, C \rightarrow I, EC \rightarrow A, EC \rightarrow B, EI \rightarrow C.$$ 

4.

(a) $IS$ is a key, i.e. a minimal superkey. Indeed, $(IS)^+ = ISDBQO$. To see that it is minimal, note that $I$ is not a key and $S$ is not a key.

(b) $IS$ is the only minimal key. To see this, it is enough to note that any key $K$ must contain $IS$. This is obvious, because neither $I$ nor $S$ appear on the right-hand side of any fd.

(c) A BCNF decomposition with lossless join obtained by the algorithm is:

$$\rho = \{SD, IB, IO, ISQ\}.$$ 

(See Figure 1.)

(d) It is easy to check that $S \rightarrow D, I \rightarrow B, IS \rightarrow Q, B \rightarrow O$ is minimal. Thus, $\{SD, IB, ISQ, BO\}$ is a 3NF decomposition which is dependency preserving (this happens to be the same as the BCNF above). Note that the key $IS$ is a subset of one relation in the decomposition ($ISQ$) so there is no need to add it, and the decomposition also has lossless join.

5.
Figure 1: BCNF decomposition in problem 5(c).

(a) By decomposing $ABCD$ using $D \rightarrow C$ (which violates BCNF within $ABCD$), we obtain $\{DC, ABD\}$. Clearly, $DC$ is in BCNF (no violation can occur in a two-attribute relation). In $ABD$, the only violations could come from fds with a single attribute on the lefthand side. Thus, it is sufficient to check $A^+, B^+$, and $D^+$ within $ABD$: $A^+ \cap ABD = A, B^+ \cap ABD = B, D^+ \cap ABD = D$. So $ABD$ is in BCNF and the final BCNF decomposition is $\{DC, ABD\}$.

(b) It is necessary to check preservation of $AB \rightarrow C$ and $B \rightarrow C$. Our algorithm shows that $AB \rightarrow C$ is preserved, but $B \rightarrow C$ is not.
(c) First, we rewrite the fds as

\[ AB \rightarrow C, AB \rightarrow D, D \rightarrow C, B \rightarrow C. \]

Clearly, \( AB \rightarrow C \) is redundant and the remainder set is minimal. Thus, the 3NF decomposition is \{\( ABD, DC, BC \)\}. Note that \( ABD \) contains the key \( AB \), so there is no need to add a key to the schema. So the above 3NF decomposition is dependency preserving and has lossless join.