6.1

```
select B1.sid, B2.sid
From catalog B1, catalog B2
where (B1.pid = B2.pid) AND (B1.Sid = B2Sid) AND
                (B1.price > B2.price)
```

6.2

```
Select S1.sid
From Supplier S1
where S1.address = 'Ruston city' OR
      S1.sid IN (select C1.sid
                 FROM part P1, catlog C1
                 where P1.color = 'red' and
                       P1.pid = C1.pid)
```

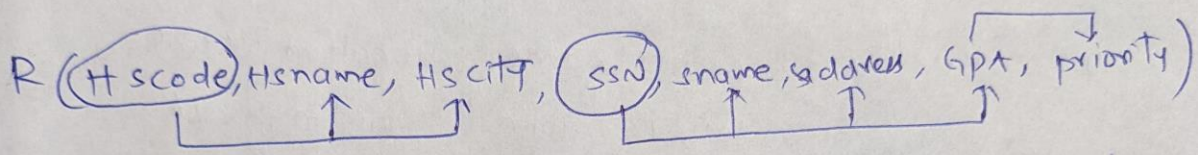
(8.1) compute the candidate keys:

Given fd; Hscode → Hsname, Hscity

Hscode⁺: Hscode, Hsname, Hscity

GPA⁺: GPA, priority

SSN⁺: SSN, sname, saddress, GPA.



SSN, Hscode⁺: sname, saddress, GPA, priority, Hscode, Hsname, Hscity.

∴ SSN, Hscode is a primary key

Hscode → Hsname, Hscity (p.d)

SSN → sname, sadress, GPA (p.d)

GPA → priority (non-prime) - violation of BCNF.

using - GPA → priority

→ R1(GPA, priority)

→ R2(GPA, HCity, Hsname, Hscode, saddress, sname, SSN)

using SSN → sname, saddress, GPA

→ R21(SSN, Sname, saddress, GPA)

→ R22(SSN, HCity, Hsname, Hscode)

using Hscode → Hsname, Hscity

R221(Hscode, Hsname, Hscity)

R222(Hscode, SSN).

So, R1(GPA, priority), R21(SSN, sname, saddress, GPA), R221(Hscode, Hsname, Hscity), R222(Hscode, SSN) are. BCNF.

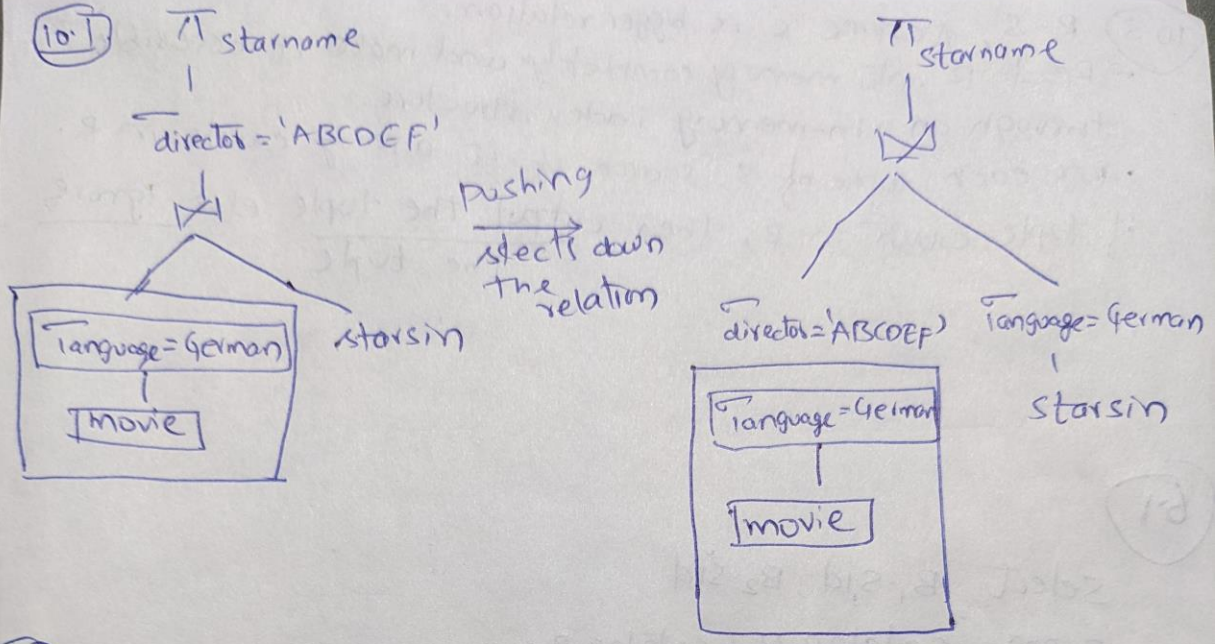
R1, R21, R221, R222

8.2

Checks on parse tree

1) Syntactic checks: is the syntax of every operator correct?
2) Entity checks: does every relation name refer to a valid relation?
3) View expansion: if a relation name refers to a view replace the relation node with the parse tree of the view
4) Attribute checks: does every attribute name refer to valid attributes?
5) Type checks: does each attribute participating in an expression have the proper type?

10.1



10.2 Compute candidate key

A^+ : ABC, A^+ = ABC, AE^+ : AEBCD,

AEF^+ : AEFBCD, AEF covers all, AEF is a candidate key

$A \to B$ (pd), $A \to C$ (pd), $AE \to D$ (pd)

using → $A \to B$
- → R_1 (AB)
- R_2 (ACDEFG)
  - using $A \to C$
    - R_{21} (AC)
    - R_{22} (ADEFG)
      - using $AE \to D$
        - R_{221} (AED)
        - R_{22} (AEFG)

R_1(ABC), R_2(AED), R_3(AEFG) are in BCNF.

10-3) R-S, Assume 'S' is bigger relation.

⇒ Read 'R' into memory completely and make it accessible through an in-memory index structure.

- For each tuple of S, search if it already exists in R.

If tuple exists in R, then output the tuple else ignore the tuple

9.1)

R

CWID	Name	Subject	location	score
1. 10156731	John	computer Nw	Ruston	96
10145221	John	DBMS	Ruston	100

R1

CWID	Name	Subject
10156731	John	CN
10145221	John	DBMS

R2

CWID	location	score
10156731	Ruston	96
10145221	Ruston	100

R1 ∪ R2

CWID	Name	subject	location	score
10156731	John	CN	Ruston	96
10145221	John	DBMS	Ruston	100

(i) $R_1 \cup R_2 = R$.

(ii) $R_1 \cap R_2$ = CWID ie $R_1 \cap R_2 \neq \phi$

(iii) $R_1 \cap R_2$ = CWID, with CWID we can access the R_1 (the relation) ie CWID, Name, subject (R_1)

The (1) st satisfies the all the three rules, it is a lossless join.

2.

R_1

CWID	Name	location	score
101673	John	Ruston	96
101H221	John	Ruston	100

R_2

location	score
Ruston	96
Ruston	100

(i) $R_1 \cup R_2$ –

CWID	Name	location	score
101673	John	Ruston	96
101H221	John	Ruston	100

$R_1 \cup R_2 \neq R$.

(ii) $R_1 \cap R_2$ =

location	score
Ruston	96
Ruston	100

$R_1 \cap R_2 \neq \phi$

(iii) $R_1 \cap R_2 = R_2$ → it give R_2,

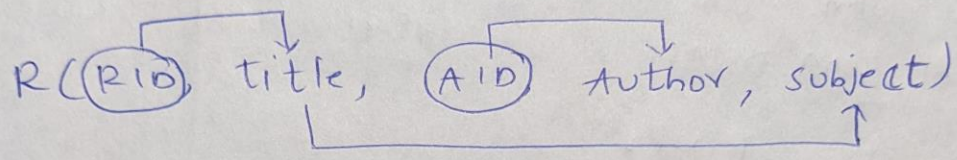
The relations must and should satisfy the three rules to become a lossless join, so the above (2) doesn't satisfy, so it is a lossy join.

∴ 1 is a lossless and 2 is a lossy join

(9.2) Report(RID, title, AID, Author, subject)

RID → title,
title → subject
AID → Author

So Ans: edge diagram to find the candidate key



Closure of RID, AID$^+$: RID, AID, Author, Title, subject

RID, AID is a candidate key.

RID → title (partial dependency)
title → subject (transitive dependency)
AID → Author (partial dependency)

R(RID, Title, AID, Author, subject) → R_1(RID, AID)
→ R_2(RID, title, subject)
→ R_{21}(RID, title)
→ R_{22}(title, subject)
→ R_3(AID, Author)

R_1, R_{21}, R_{22}, R_3 are in 3NF.