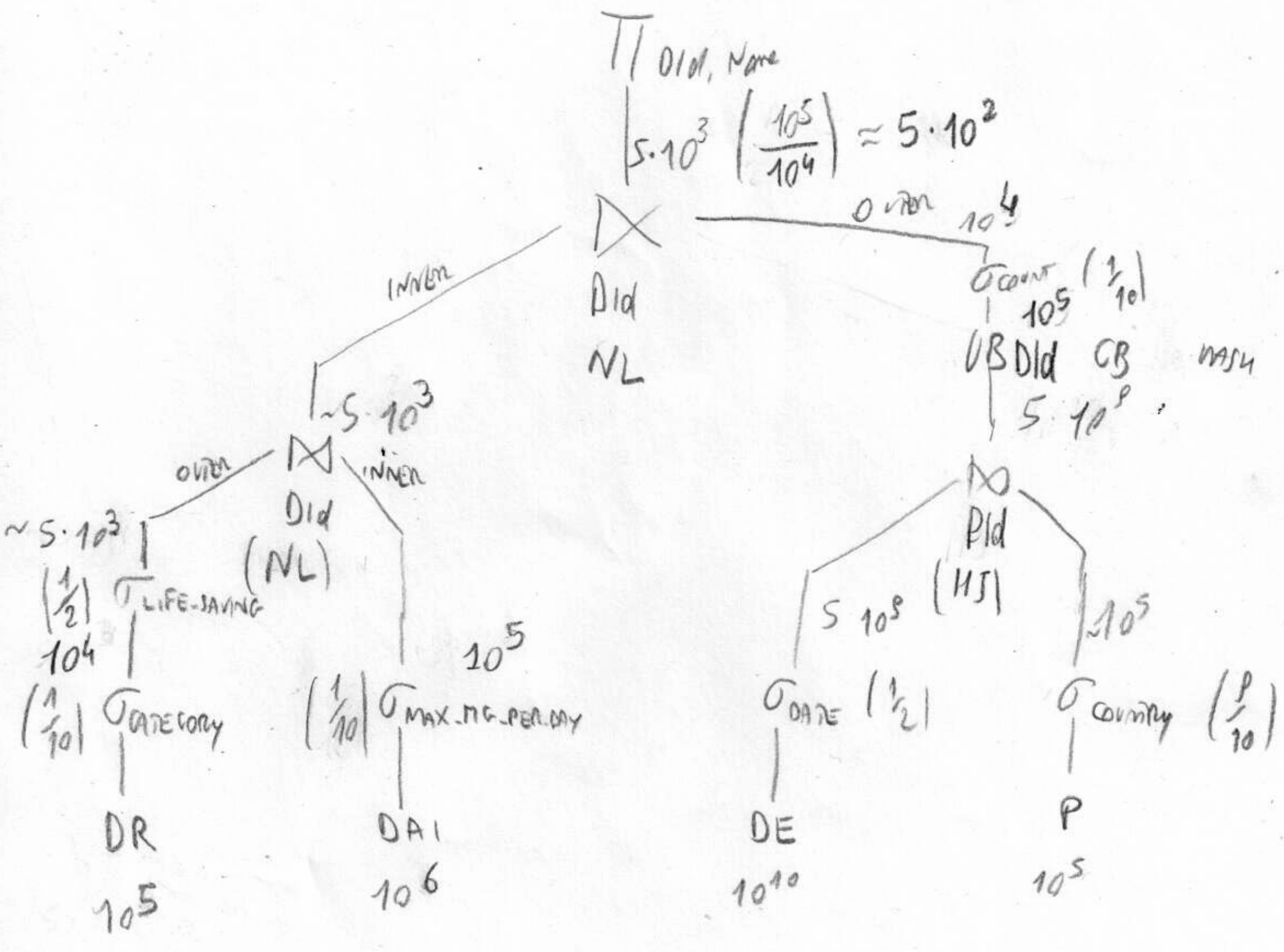


Indices

DR (Category) : SEC HASH DAI (MAX_MG_PER_DAY) : SEC HASH

AP without indices

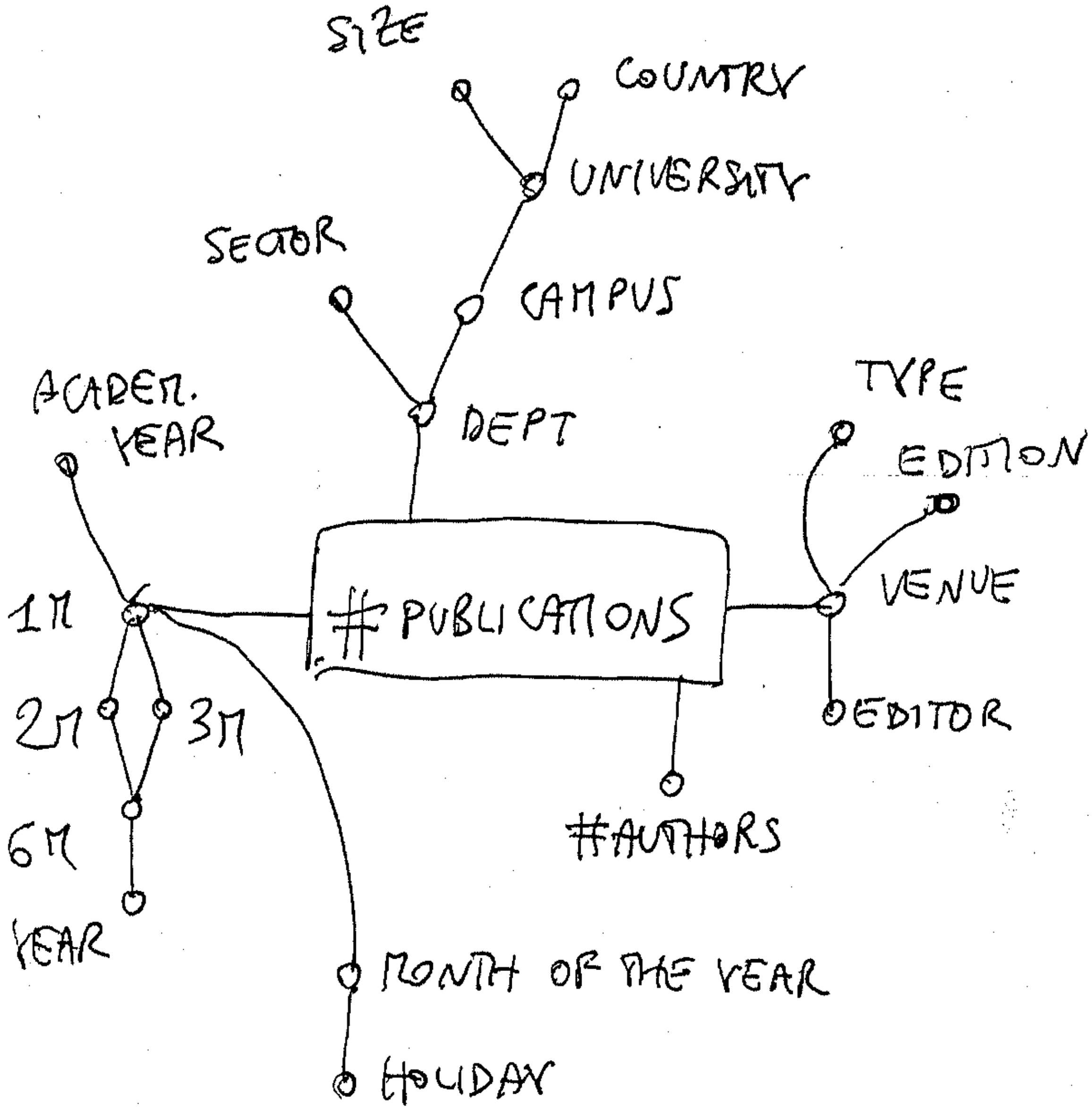
TABLE ACCESS FULL + FILTER (DR, DAI, DE, P)

AP with indices

INDEX RANGE SCAN + ACCESS BY ROWID ON DR AND DAI

```
create or replace trigger END_RENTAL
after update of EndRentalTimeStamp on RENTAL
for each row
when (old.EndRentalTimeStamp is NULL and new.EndRentalTimeStamp is not NULL)
declare
myDurationUse, myDurationStop NUMBER;
myCostUse, myCostStop NUMBER;
TotAmount NUMBER;

begin

update CAR
set CurrentState = 'free'
where Plate = :new.Plate;

update RENTAL_ACTIVITY
set EndRentalTimeStamp = :new.EndRentalTimeStamp
where Plate = :new.Plate and EndRentalTimeStamp is NULL;

select SUM(EndRentalTimeStamp-StartRentalTimeStamp) into myDurationUse
from RENTAL_ACTIVITY
where Plate = :new.Plate and TypeOfActivity = 'use';

select CostPerMinute into myCostUse
from RENTAL_RATE
where TypeOfActivity = 'use';

select SUM(EndRentalTimeStamp-StartRentalTimeStamp) into myDurationPark
from RENTAL_ACTIVITY
where Plate = :new.Plate and TypeOfActivity = 'park';

if (myDurationPark is NULL) then
  myDurationPark := 0;
else
   select CostPerMinute into myCostPark
   from RENTAL_RATE
   where TypeOfActivity = 'park';
enf if;

TotAmount = TO_MINUTES(myDurationUse)*myCostUse+ TO_MINUTES(myDurationPark)*myCostPark;

select MAX(ReceiptCode) into myCod
from RENTAL_RECEIPT;

if (myCod is NULL) then
   myCod := 0;
end if;

insert into RENTAL_RECEIPT (...)
values (myCod+1, :new.CustomerCode, :new.Plate, new.StartRentalTimeStamp,
        :new.EndRentalTimeStamp, TotAmount);
```

```
delete from RENTAL_ACTIVITY
where Plate = :new.Plate;

end;


create trigger CheckEndTimeStamp
after insert on RENTAL_ACTIVITY
for each row
when (NEW.EndTimeStamp is not null)

begin
raise_application_error(...)
end;
```



2016-01-27
SGBD

```
QUERY

(A)

SELECT UNIVERSITY, TYPE, YEAR,
SUM(PUB) / COUNT( DISTINCT(MONTH) ) AS
MONTHLY_AVG,
SUM(SUM(PUB)) OVER (PARTITION BY UNIVERSITY, TYPE
ORDER BY YEAR,
ROWS UNBOUNDED PRECEDING )

FROM <TABLES>

WHERE <JOINS>
GROUP BY UNIVERSITY, TYPE, YEAR

QUERY

(B)

SELECT YEAR, DEPT, (UNIVERSITY)
100 * SUM(PUB) / SUM(SUM(PUB)) OVER
(PARTITION BY YEAR, UNIVERSITY),
RANK() OVER (PARTITION BY DEPT
ORDER BY SUM(PUB) DESC)

FROM <TABLES>

WHERE <JOINS>

GROUP BY YEAR, DEPT, UNIVERSITY
```