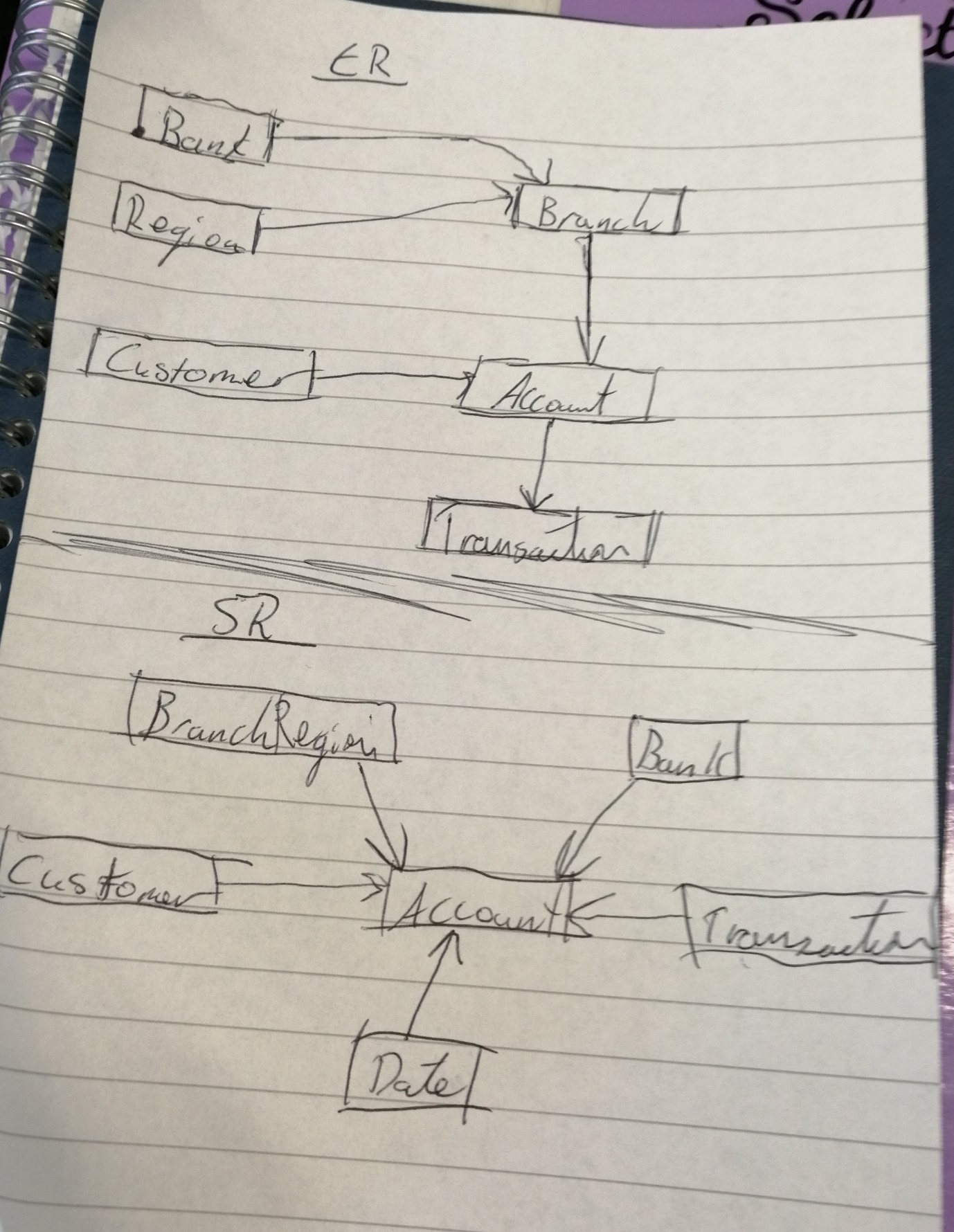
Lab 5

# Problem 1



Er diagram

-----------------

bank - bankid, name

Region - regionid, regionname, country

Customer - ccode, name, address, phone, bday, gender, salary

branch - branchid, name, address, regionid (fk), bankid (fk)

Account - aCode, openingdate, balance, branchid (fk), accounttype, ccode (fk)

tranaction - tcode, date, type, amount, aCode (fk)

SR Diagram

-------------------

Date - DateId, DayDate, DayDate\_YYYYMMDD, DayOfWeekName ,DayOfWeekNameAbbrv ,DayNumberInWeek ,DayNumberInMonth

,DayNumberInQuarte ,DayNumberInYear ,WeekDayIndicator ,WeekEndIndicator ,Week\_YYYYWW ,WeekNumberInYear ,Month\_YYYYMM ,MonthName ,MonthNameAbbrv

,MonthNumberInYear ,Quarter\_YYYYQ ,QuarterName ,QuarterNameAbrv ,QuarterNumberInYear ,Year

bank - bankid, name

BranchRegion - Region - regionid, regionname, country, branchid, name, address

Customer - ccode, name, address, phone, bday, gender, salary

tranaction - tcode, date, type, amount, aCode (fk)

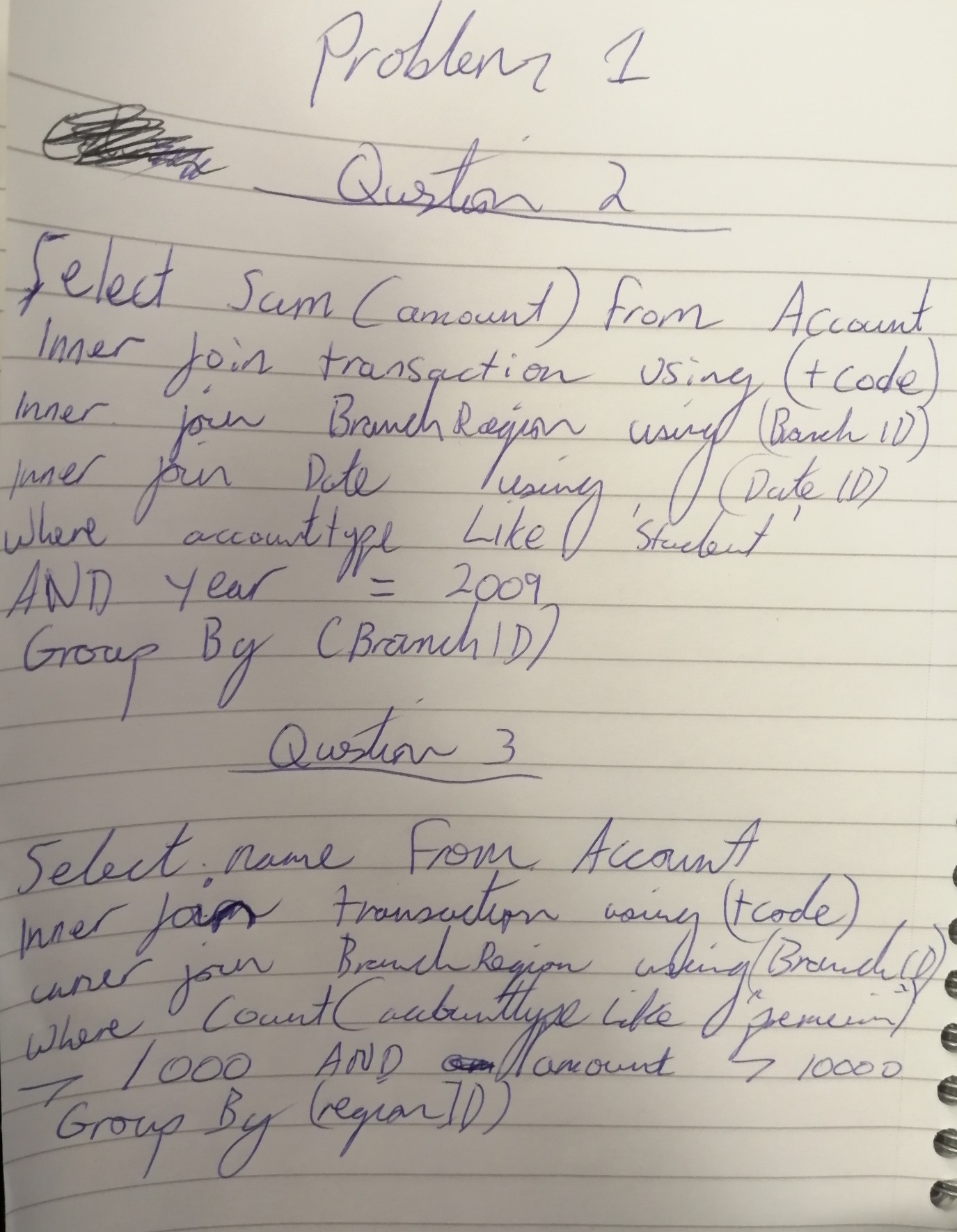
Account - aCode, openingdate, balance, branchid (fk), accounttype, ccode (fk), bankid (fk), tcode (fk)

## Explanation

The Account is the main part of the schema due to the connection to each part of the application. So it would make sense to set this part to the fact of the application. Another key decision would be the combination of this to be the most descriptive of the model we are trying to implement and the fact most banks use accounts to identify their users as business requirements.

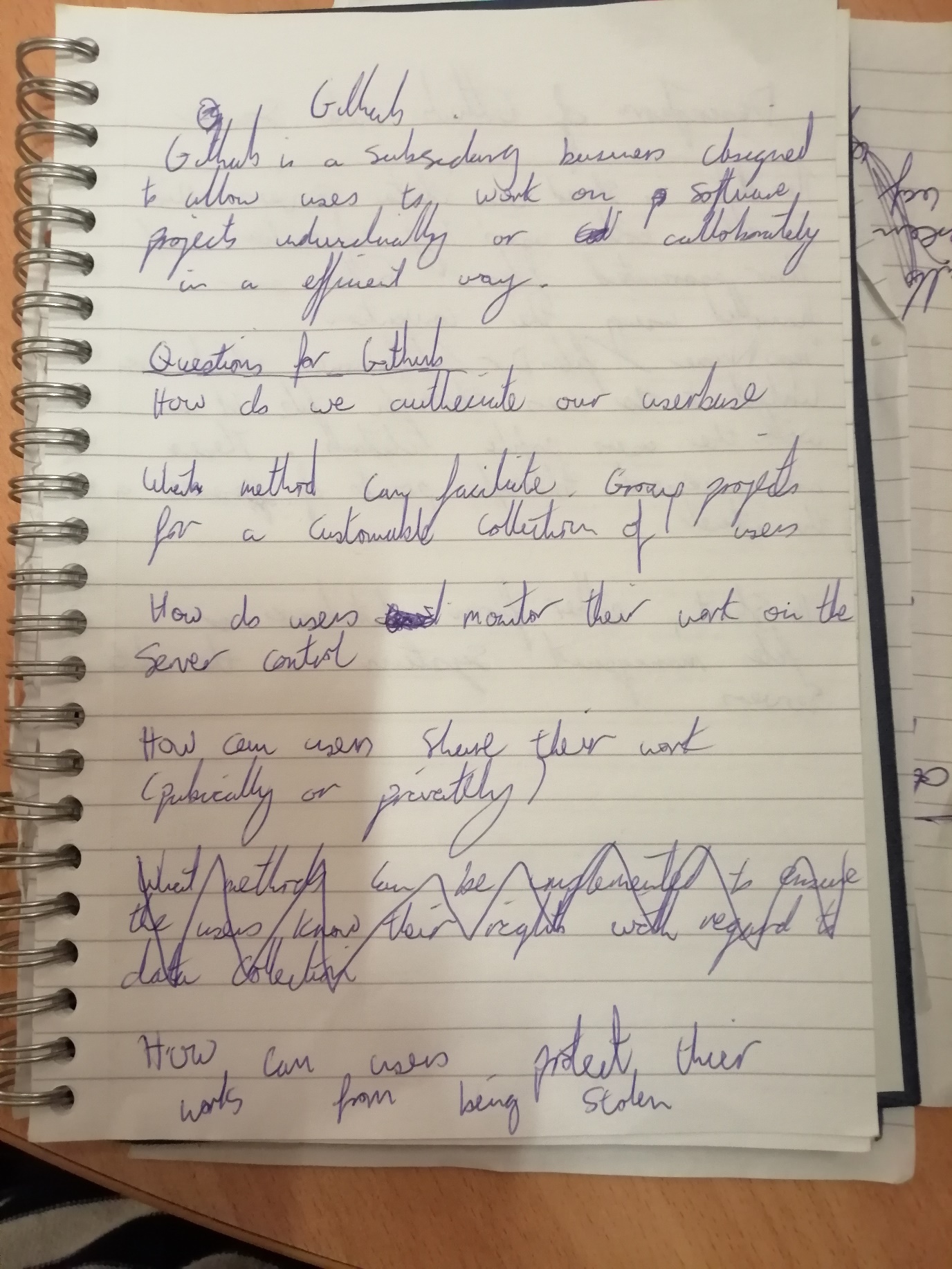
All the other parts are additional dimensions to the fact as they provide a way to deploy a more comprehensive study of the schema. However, they have a dependency of the fact to actual provide description suitable for analysis. Otherwise they only provide vague or simply solutions.

The account also contained the highest granularity (GRAIN) level, which is another reason to set it as the fact of the schema.

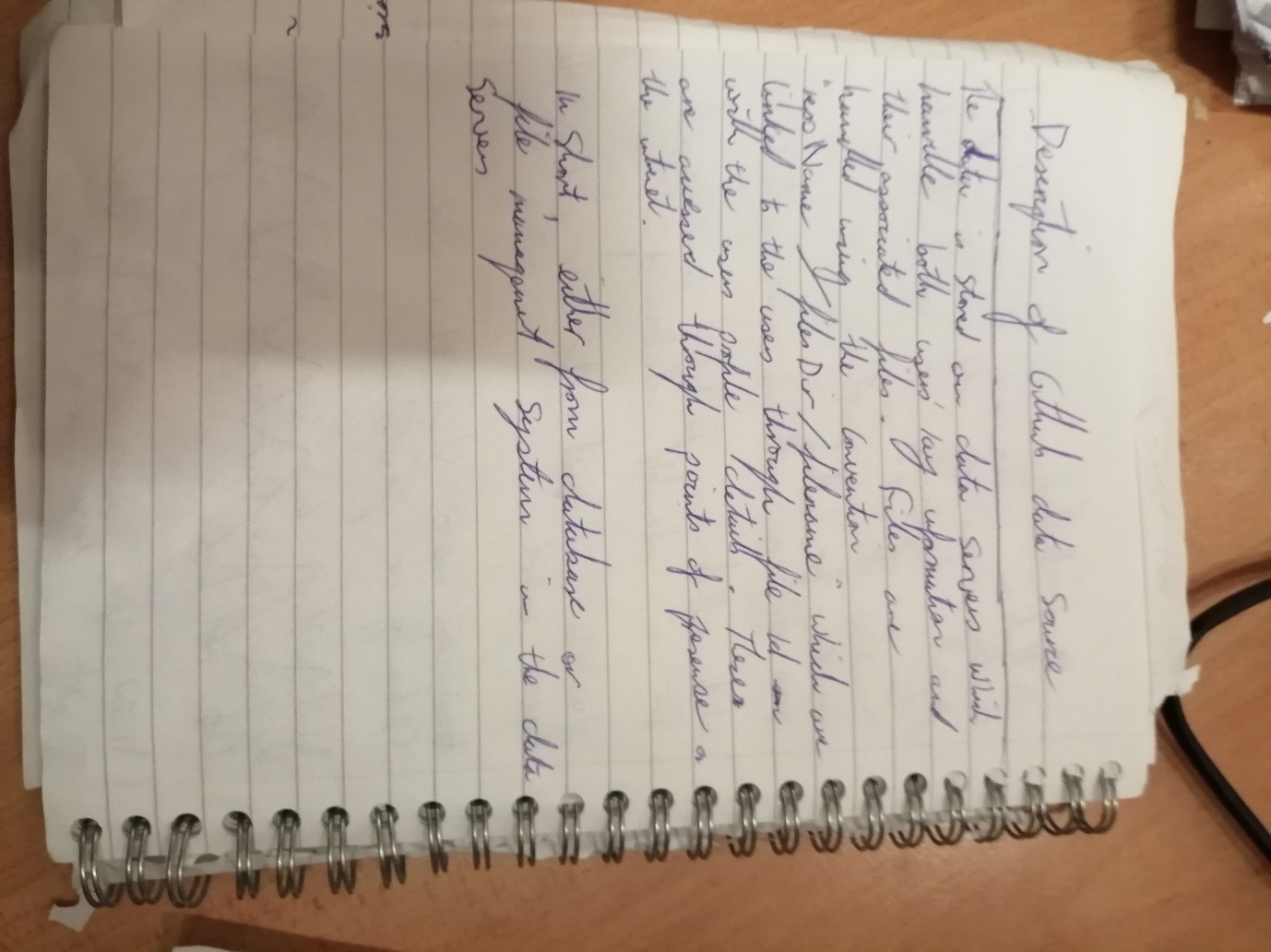


# Problem 2

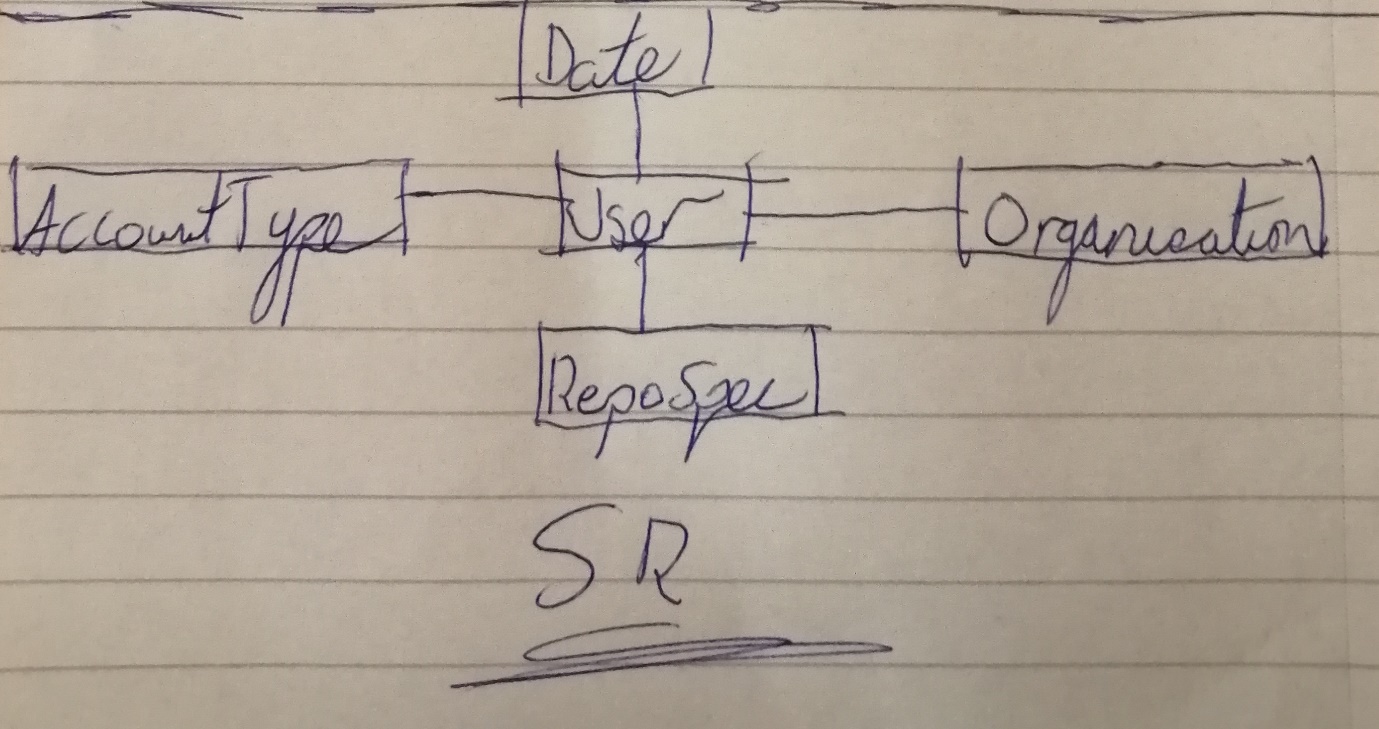
Part 1



Part 2



Part 3



Date - DateId, DayDate, DayDate\_YYYYMMDD, DayOfWeekName ,DayOfWeekNameAbbrv

,DayNumberInWeek ,DayNumberInMonth ,DayNumberInQuarte ,DayNumberInYear

,WeekDayIndicator ,WeekEndIndicator ,Week\_YYYYWW ,WeekNumberInYear

,Month\_YYYYMM ,MonthName ,MonthNameAbbrv ,MonthNumberInYear ,Quarter\_YYYYQ ,

QuarterName ,QuarterNameAbrv ,QuarterNumberInYear ,Year

AccountType - UserName, AccountType, AccountTypeID, isBillable,

billingInterval, IsOfferAvail

Organisation - UserName, OrganisationName, OrganisationID, CreatedBy,

OrganisationRoleID, OrganisationRole, RolePermissions, RepoID,

InnerOrganisationID, RoleDateCreated,RoleCreatedBy

RepoSpec - RepoName, RepoURL, RepoID, fileName, fileID,

fileVersion, OrganisationID, AccountTypeID, IsPrivate,

DateCreated, RepoDescription, UserName, RepoRefID, RepoDetailsID

User - Username, Name, email, Password, UserVerified, Description,

AccountTypeID, RepoID, OrganisationID, DateCreated, DateID

--answer business requirement 1,4,5

SELECT \* FROM USER

INNER JOIN AccountType USING AccountTypeID

INNER JOIN RepoSpec USING RepoID

--check if the user has authecaite their account

WHERE UserVerified == 1

--Check if there exists a repo, allowing the possibility of sharing work

AND RepoName IS NOT NULL

--check if the work can be protected

AND AccountType.lower() LIKE 'STUDENT' OR AccountType.lower() LIKE 'ENTERPRISE'

Part 4

--Answer 2 and 3

SELECT \* FROM USER

INNER JOIN RepoSpec USING RepoID

INNER JOIN Organisation USING OrganisationID

--CHECK if there exists work that we can monitor

WHERE fileID IS NOT NULL

--Checking if there is a organisation that exists, allowing for callboratition

AND OrganisationRoleID IS NOT NULL