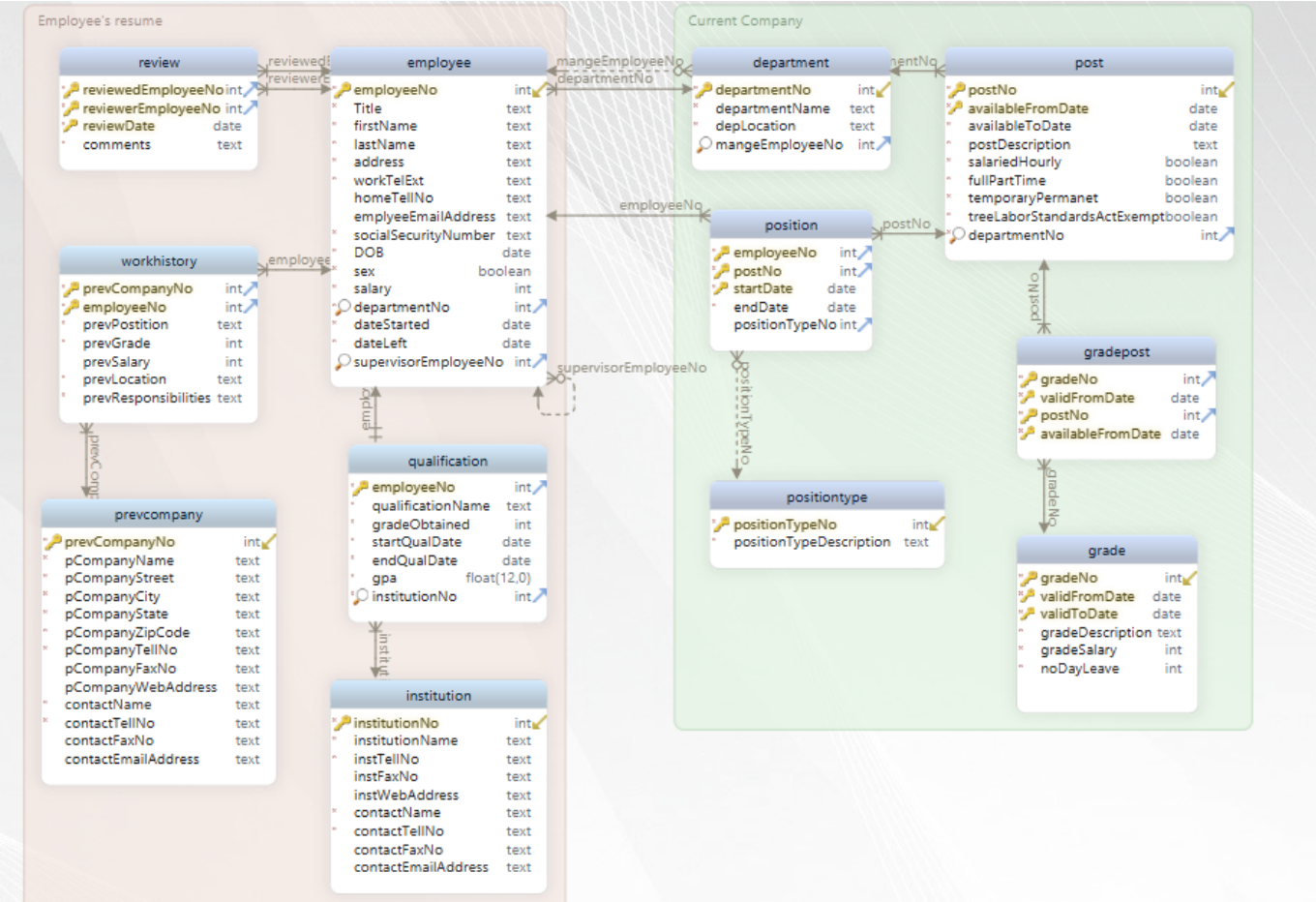
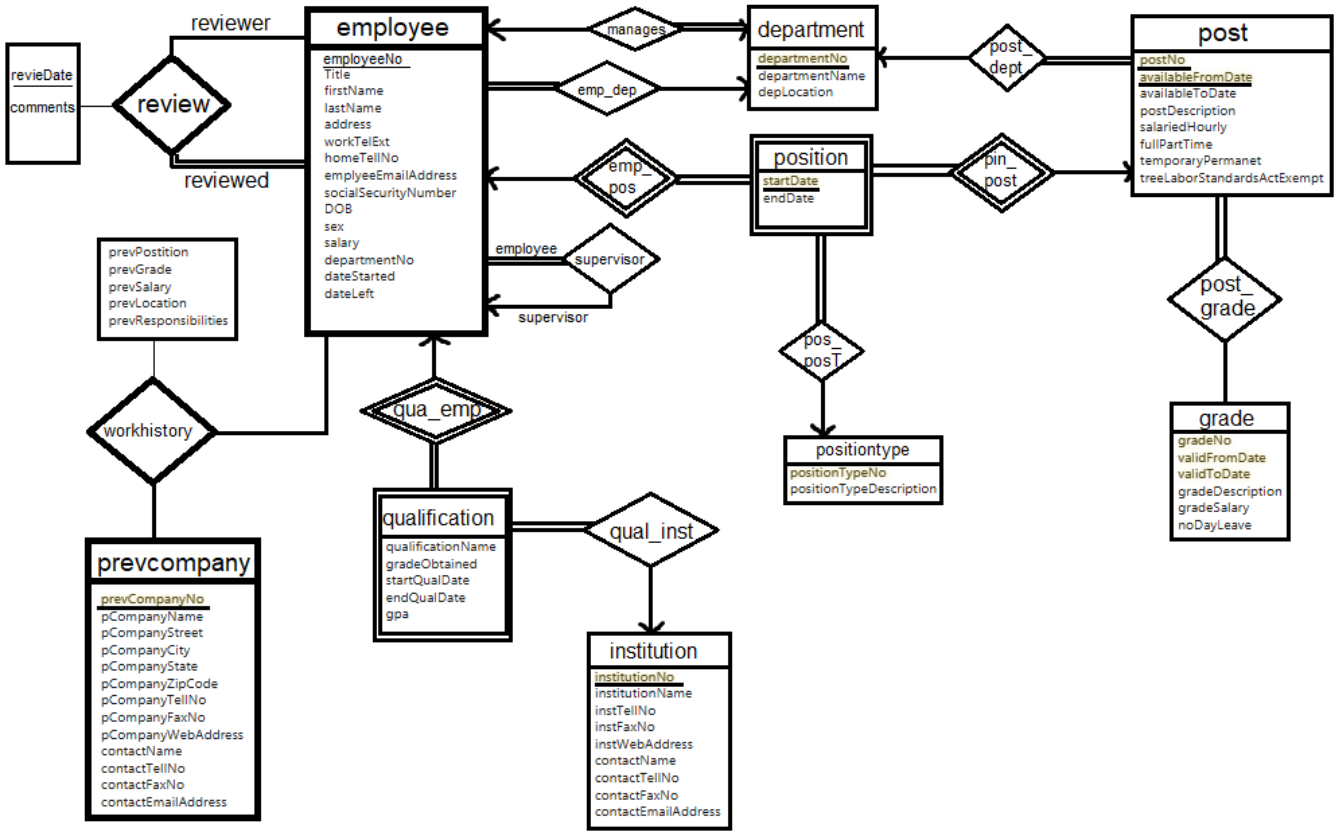
Database Project

we established our database on sql file(created tables, integrity constraint, forigne keies, ...) named "project.sql" and inserted the data we needed into "data-insertion".
so our relational schema is gonna be like:



and also ER model will be like:



now we want to perform some queries using select, delete, update, aggregation functions, nested, exist, comparison and membership, types of join, view and grant statementds.

Initial Tables

Grade

```
CREATE TABLE grade (
    gradeNo              int  NOT NULL    ,
    validFromDate        date  NOT NULL    ,
    validToDate          date  NOT NULL    ,
    gradeDescription     text  NOT NULL    ,
    gradeSalary          int  NOT NULL    ,
    noDayLeave           int  NOT NULL    ,
    positionTypeNo       int  NOT NULL    ,
    CONSTRAINT pk_grade PRIMARY KEY ( gradeNo, validFromDate, validToDate
),
    CONSTRAINT unq_grade_gradeno UNIQUE ( gradeNo )
 );

 CREATE INDEX fk_grade_positiontype ON grade ( positionTypeNo );

ALTER TABLE gradepost ADD FOREIGN KEY ( gradeNo ) REFERENCES grade( gradeNo
) ON DELETE RESTRICT ON UPDATE RESTRICT;
```

Institution

```
CREATE TABLE institution (
    institutionNo        int  NOT NULL    PRIMARY KEY,
    institutionName      text  NOT NULL    ,
    instTellNo           text  NOT NULL    ,
    instFaxNo            text      ,
    instWebAddress       text      ,
    contactName          text  NOT NULL    ,
    contactTellNo        text  NOT NULL    ,
    contactFaxNo         text      ,
    contactEmailAddress  text
);

ALTER TABLE qualification ADD FOREIGN KEY ( institutionNo ) REFERENCES
institution( institutionNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;
```

PositionType

```
CREATE TABLE positiontype (
    positionTypeNo       int  NOT NULL    PRIMARY KEY,
    positionTypeDescription text  NOT NULL
 );

ALTER TABLE position ADD FOREIGN KEY ( positionTypeNo ) REFERENCES
positiontype( positionTypeNo ) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

PrevComapny

```
CREATE TABLE prevcompany (
    prevCompanyNo        int  NOT NULL    PRIMARY KEY,
    pCompanyName         text  NOT NULL    ,
    pCompanyStreet       text  NOT NULL    ,
    pCompanyCity         text  NOT NULL    ,
    pCompanyState        text  NOT NULL    ,
    pCompanyZipCode      text  NOT NULL    ,
    pCompanyTellNo       text  NOT NULL    ,
    pCompanyFaxNo        text      ,
    pCompanyWebAddress   text      ,
    contactName          text  NOT NULL    ,
    contactTellNo        text  NOT NULL    ,
    contactFaxNo         text      ,
    contactEmailAddress  text
 );

ALTER TABLE workhistory ADD FOREIGN KEY ( prevCompanyNo ) REFERENCES
prevcompany( prevCompanyNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;
```

Department

```sql
CREATE TABLE department (
    departmentNo         int  NOT NULL    PRIMARY KEY,
    departmentName       text  NOT NULL    ,
    depLocation          text  NOT NULL    ,
    mangeEmployeeNo      int
 );

CREATE INDEX mangeEmployeeNo ON department ( mangeEmployeeNo );

ALTER TABLE employee ADD FOREIGN KEY ( departmentNo ) REFERENCES
department( departmentNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE post ADD FOREIGN KEY ( departmentNo ) REFERENCES department(
departmentNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;
```

Employee

```sql
CREATE TABLE employee (
    employeeNo           int  NOT NULL    PRIMARY KEY,
    Title                text  NOT NULL    ,
    firstName            text  NOT NULL    ,
    lastName             text  NOT NULL    ,
    address              text  NOT NULL    ,
    workTelExt           text  NOT NULL    ,
    homeTellNo           text      ,
    emplyeeEmailAddress  text      ,
    socialSecurityNumber text  NOT NULL    ,
    DOB                  date  NOT NULL    ,
    sex                  boolean  NOT NULL    ,
    salary               int  NOT NULL    ,
    departmentNo         int  NOT NULL    ,
    dateStarted          date  NOT NULL    ,
    dateLeft             date  NOT NULL    ,
    supervisorEmployeeNo int
 );

CREATE INDEX departmentNo ON employee ( departmentNo );

CREATE INDEX fk_employee_employee ON employee ( supervisorEmployeeNo );

ALTER TABLE department ADD FOREIGN KEY ( mangeEmployeeNo ) REFERENCES
employee( employeeNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE employee ADD FOREIGN KEY ( supervisorEmployeeNo ) REFERENCES
employee( employeeNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE position ADD FOREIGN KEY ( employeeNo ) REFERENCES employee(
employeeNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE qualification ADD FOREIGN KEY ( employeeNo ) REFERENCES
```

```
employee( employeeNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE review ADD FOREIGN KEY ( reviewedEmployeeNo ) REFERENCES
employee( employeeNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE review ADD FOREIGN KEY ( reviewerEmployeeNo ) REFERENCES
employee( employeeNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE workhistory ADD FOREIGN KEY ( employeeNo ) REFERENCES employee(
employeeNo ) ON DELETE RESTRICT ON UPDATE RESTRICT;
```

GradePost

```
CREATE TABLE gradepost (
    gradeNo              int  NOT NULL    ,
    validFromDate        date  NOT NULL    ,
    postNo               int  NOT NULL    ,
    availableFromDate    date  NOT NULL    ,
    CONSTRAINT pk_gradepost PRIMARY KEY ( gradeNo, validFromDate, postNo,
availableFromDate )
 );

CREATE INDEX fk_gradepost_post ON gradepost ( postNo );
```

Position

```
CREATE TABLE position (
    employeeNo           int  NOT NULL    ,
    postNo               int  NOT NULL    ,
    startDate            date  NOT NULL    ,
    endDate              date  NOT NULL    ,
    positionTypeNo       int      ,
    CONSTRAINT pk_position PRIMARY KEY ( employeeNo, postNo, startDate )
 );

CREATE INDEX employeeNo ON position ( employeeNo, postNo );

CREATE INDEX postNo ON position ( postNo );

CREATE INDEX fk_position_positiontype ON position ( positionTypeNo );
```

Post

```
CREATE TABLE post (
    postNo               int  NOT NULL    ,
    availableFromDate    date  NOT NULL    ,
```

```
    availableToDate      date  NOT NULL    ,
    postDescription      text  NOT NULL    ,
    salariedHourly       boolean  NOT NULL    ,
    fullPartTime         boolean  NOT NULL    ,
    temporaryPermanet    boolean  NOT NULL    ,
    treeLaborStandardsActExempt boolean  NOT NULL    ,
    departmentNo         int  NOT NULL    ,
    CONSTRAINT pk_post PRIMARY KEY ( postNo, availableFromDate ),
    CONSTRAINT postNo UNIQUE ( postNo )
 );

CREATE INDEX departmentNo ON post ( departmentNo );

ALTER TABLE gradepost ADD FOREIGN KEY ( postNo ) REFERENCES post( postNo )
ON DELETE RESTRICT ON UPDATE RESTRICT;

ALTER TABLE position ADD FOREIGN KEY ( postNo ) REFERENCES post( postNo )
ON DELETE RESTRICT ON UPDATE RESTRICT;
```

Qualification

```
CREATE TABLE qualification (
    employeeNo           int  NOT NULL    PRIMARY KEY,
    qualificationName    text  NOT NULL    ,
    gradeObtained        int  NOT NULL    ,
    startQualDate        date  NOT NULL    ,
    endQualDate          date  NOT NULL    ,
    gpa                  float(12,0)  NOT NULL    ,
    institutionNo        int  NOT NULL
 );

CREATE INDEX institutionNo ON qualification ( institutionNo );
```

Review

```
CREATE TABLE review (
    reviewedEmployeeNo   int  NOT NULL    ,
    reviewerEmployeeNo   int  NOT NULL    ,
    reviewDate           date  NOT NULL    ,
    comments             text  NOT NULL    ,
    CONSTRAINT pk_review PRIMARY KEY ( reviewedEmployeeNo,
reviewerEmployeeNo, reviewDate )
 );

CREATE INDEX reviewedEmployeeNo ON review ( reviewedEmployeeNo,
reviewerEmployeeNo );

CREATE INDEX reviewerEmployeeNo ON review ( reviewerEmployeeNo );
```

WorkHistory

```
CREATE TABLE workhistory (
    prevCompanyNo        int  NOT NULL    ,
    employeeNo           int  NOT NULL    ,
    prevPostition        text  NOT NULL    ,
    prevGrade            int  NOT NULL    ,
    prevSalary           int      ,
    prevLocation         text  NOT NULL    ,
    prevResponsibilities text  NOT NULL    ,
    CONSTRAINT pk_workhistory PRIMARY KEY ( prevCompanyNo, employeeNo )
 );

CREATE INDEX fk_workhistory_employee ON workhistory ( employeeNo );
```

Queries

consider this relation instance on qualification:

employeeNo	qualificationName	gradeObtained	startQualDate	endQualDate	gpa	institutionNo
0	Abel	1,570,126,089	15.08.2014	12.08.2014	730,520,027,136.00	7
1	Colby	1,206,768,426	14.09.2002	29.06.2011	283,149,008,896.00	7
2	Robbie	2,120,470,094	21.08.2011	09.05.2004	697,711,984,640.00	7
3	Kristen	573,006,458	12.11.2008	10.08.2005	583,177,011,200.00	3
4	Carla	489,965,424	13.05.2015	04.08.2013	277,766,995,968.00	1
5	Heath	160,618,523	03.07.2011	08.09.2005	759,961,026,560.00	2
6	Erick	1,935,844,031	01.04.2011	24.02.2018	88,258,396,160.00	3
7	Lawanda	1,840,659,869	05.05.2001	16.05.2018	448,598,016,000.00	7
8	Janice	1,066,918,395	02.06.2001	17.05.2018	488,905,015,296.00	5
9	Keri	876,785,566	05.08.2000	06.02.2016	392,875,999,232.00	8

as you can see, the gpa scale is too large and covers the range of 1 billion to 1000 billion, thus if we divide it to 50 billion, it's range will be from 1 to 20, wich is more formal.

```
UPDATE qualification
SET gpa = gpa / 50000000000
```

employeeNo	qualificationName	gradeObtained	startQualDate	endQualDate	gpa	institutionNo
0	Abel	1,570,126,089	15.08.2014	12.08.2014	15.00	7
1	Colby	1,206,768,426	14.09.2002	29.06.2011	6.00	7
2	Robbie	2,120,470,094	21.08.2011	09.05.2004	14.00	7
3	Kristen	573,006,458	12.11.2008	10.08.2005	12.00	3
4	Carla	489,965,424	13.05.2015	04.08.2013	6.00	1
5	Heath	160,618,523	03.07.2011	08.09.2005	15.00	2
6	Erick	1,935,844,031	01.04.2011	24.02.2018	2.00	3
7	Lawanda	1,840,659,869	05.05.2001	16.05.2018	9.00	7
8	Janice	1,066,918,395	02.06.2001	17.05.2018	10.00	5
9	Keri	876,785,566	05.08.2000	06.02.2016	8.00	8

now assume that we want delete employee number 4 from this table, therfore:

```
DELETE FROM qualification
WHERE employeeNo = 4
```

employeeNo	qualificationName	gradeObtained	startQualDate	endQualDate	gpa	institutionNo
0	Abel	1,570,126,089	15.08.2014	12.08.2014	15.00	7
1	Colby	1,206,768,426	14.09.2002	29.06.2011	6.00	7
2	Robbie	2,120,470,094	21.08.2011	09.05.2004	14.00	7
3	Kristen	573,006,458	12.11.2008	10.08.2005	12.00	3
5	Heath	160,618,523	03.07.2011	08.09.2005	15.00	2
6	Erick	1,935,844,031	01.04.2011	24.02.2018	2.00	3
7	Lawanda	1,840,659,869	05.05.2001	16.05.2018	9.00	7
8	Janice	1,066,918,395	02.06.2001	17.05.2018	10.00	5
9	Keri	876,785,566	05.08.2000	06.02.2016	8.00	8

1. Info on employees whose salary is above salary average

```
SELECT employeeNo, firstName, lastName
FROM employee
WHERE salary > (
    SELECT avg(salary)
    FROM employee
);
```

	employeeNo	firstName	lastName
1	0	Latoya	Cisneros
2	2	Lakeisha	Byrd
3	7	Hector	Daniels
4	9	Mike	Middleton
5	11	Ruby	Garrett
6	14	Harry	Ellison
7	15	Luz	Ryan
8	16	Salvatore	Matthews
9	20	Sarah	Hayes
10	21	Gavin	Combs

2. List of employees number and previous companies in which employees of department number 6 have worked in

```
    SELECT employeeNo, pCompanyName
    FROM ( employee INNER JOIN workhistory USING(employeeNo) ) INNER
JOIN prevcompany USING(prevCompanyNo)
    WHERE departmentNo = 6;
```

	* employeeNc int	pCompanyName
1	7	Rapzapin Direct Company
2	7	Klibanollover WorldWide Company
3	7	Inhupimentor WorldWide Company
4	16	Klinipentor WorldWide
5	17	Klibanollover WorldWide Company
6	25	Klierplar WorldWide Group
7	25	Zeerobedor WorldWide Corp.
8	30	Klinipentor WorldWide
9	54	Rapzapin Direct Company
10	54	Klinipentor WorldWide

3. Info on female employees whose salary is more than at least 1 male employee

```
SELECT employeeNo , firstName, lastName
FROM employee e1
WHERE e1.sex=0 AND e1.salary > SOME (
    SELECT salary
    FROM employee e2
    WHERE e2.sex = 1
);
```

	* employeeNo int	* firstName text(65535)	* lastName text(65535)
1	0	Latoya	Cisneros
2	4	Alice	Ingram
3	6	Howard	White
4	7	Hector	Daniels
5	9	Mike	Middleton
6	13	Duane	Moreno
7	14	Harry	Ellison
8	15	Luz	Ryan
9	17	Byron	Newman
10	19	Anthony	Hoffman

4. Info on employees who have worked both in "Rapzapin Direct Comapny" and "Klibanollover WorldWide Company" companies

```
    with emp_work_prev(employeeNo, firstName, lastName, pCompanyName)
AS (
        SELECT employeeNo, firstName, lastName, pCompanyName
        FROM employee INNER JOIN (workhistory INNER JOIN prevcompany
USING(prevCompanyNo)) USING(employeeNo)
    )
    SELECT employeeNo, firstName, lastName
    FROM emp_work_prev e1
    WHERE e1.pCompanyName='Rapzapin Direct Company' AND EXISTS (
        SELECT *
        FROM emp_work_prev e2
        WHERE e1.employeeNo = e2.employeeNo AND
e2.pCompanyName='Klibanollover WorldWide Company');
```

	* employeeNo int	* firstName text(65535)	* lastName text(65535)
1	7	Hector	Daniels

5. Existing post numbers at all grades in the database

```
SELECT gradeNo, postNo
FROM grade LEFT OUTER JOIN gradepost USING(gradeNo);
```

	* gradeNo int	postNo
1	0	(NULL)
2	1	0
3	2	9
4	3	8
5	4	(NULL)
6	5	9
7	5	8
8	6	4
9	6	0
10	7	9
11	7	7
12	8	2
13	9	(NULL)

6. Consider you need to permit a secretary in the department 'Abel', access on reading user information. But you only want to give her permission in her own division

```
CREATE VIEW Abel_employees AS (
SELECT e.*
FROM employee e NATURAL JOIN department
WHERE departmentName='Abel'
);
```

	* employeeNc int	* Title text(65535)	* firstName text(65535)	* lastName text(65535)	* address text(65535)	* workTelExt text(65535)	homeTellNo text(65535)	emplyeeEmailAddress text(65535)	* socialSecurityNumbe text(65535)	* DOB date	* sex tinyint	* sal in
1	1	Miss	Casey	Blackburn	814 South Rocky Secon	036-261-4538	(NULL)	vryhp.catwtl@example.c	608-75-8279	2005-01-20	1	91772
2	5	Miss.	Bernard	Neal	848 South Rocky Cowle	813-571-3331	374-263-1126	eybo.fonmsglfku@exam	865-87-9562	2017-01-27	1	93436
3	8	Mr	Vernon	Barrera	620 North Green New B	173-658-9112	879-332-6669	gwub.wbij@example.co	748-23-8076	2018-08-03	1	76293
4	11	Dr	Ruby	Garrett	882 West White Milton S	586-554-9218	343-817-1326	winf@example.com	724-16-1646	2007-01-18	1	14157
5	34	Dr.	Jeremiah	O'Neal	73 North Green Hague A	771-581-8418	041-666-2763	qlio@example.com	118-64-0691	2008-03-10	0	10063
6	36	Dr.	Gilbert	Fletcher	862 South Green Milton	334-228-5318	639-011-6427	ucwu@example.com	367-39-8734	2002-05-08	1	83289
7	47	Dr	Mickey	Hatfield	243 North Rocky Oak Av	714-415-8871	737-531-2601	pqlzgl@example.com	626-70-4288	2013-01-03	1	88815

7. Now if this secretary has a username called 'emmy', we wanna establish:

```
GRANT SELECT on Abel_employees to (emmy);
```

8. What post each employee has, is a useful query; so we create a view of each employee's info and working post

```
    CREATE VIEW employee_post AS (
        SELECT employeeNo, firstName, lastName, postNo,
postDescription
        FROM (employee NATURAL JOIN position) INNER JOIN post
USING(postNo)
    );
```

	* employeeNc int	* firstName text(65535)	* lastName text(65535)	* postNo int	* postDescription text(65535)
1	2	Lakeisha	Byrd	0	null
2	6	Howard	White	1	null
3	7	Hector	Daniels	2	null
4	4	Alice	Ingram	3	null
5	7	Hector	Daniels	4	null
6	8	Vernon	Barrera	4	null
7	5	Bernard	Neal	6	null
8	0	Latoya	Cisneros	7	null
9	0	Latoya	Cisneros	7	null
10	4	Alice	Ingram	9	null

9. Suppose a secretary with the username Monika is in charge of submitting interview records, so she must be granted access on creating records regarding reviews

```
GRANT INSERT ON review TO (monika);
```