Databases and Information Systems COMP20070

Name: ***Ahmed Jouda***

I created a database that tackles the scenario “Hospitals advertise positions which require specific skills (e.g., nursing, administrative, etc.). Candidates may be invited to interviews for the positions”.

To start off with I used the conceptual and logical design methodology to visualise the database. First I rewrote the scenario in simpler terms, then I established the main concepts (Entities and Relationships) and decided which of them should be made tables. Then I decided on the columns needed in each table. After that I assigned the primary keys. Then I assigned the forgein keys.

Rewritten Scenario:
hospitals advertise *positions* that *candidates* apply for
based on *skills* required/possessed.

Main concepts:
1. Hospital details: hospital id **(PK)**, hospital name, address, telephone number
2. Candidate details: candidate id **(PK)**, first name, surname, address, telephone number
3. CandidateSkills: id candidate, skill possessed **(BOTH PK)**
4. Position details: position id **(PK)**, type, hospital
5. PositionSkills: id positions, skill required **(BOTH PK)**
6. Interview: interview id**(PK)**, candidate number, hospital number, position number, date, jobOffered
7. Skills:skill id**(PK)**, skill name

To visualise the forgien keys I drew a rough diagram on paper and linked up the needed columns, then I implemented it on Workbench.

I could have done it without a skills table however including a skills table makes it more efficient as instead of having to search for an entire varchar when looking for a skill you can just search for the skillid. This way you can also have a reservoir of skills.

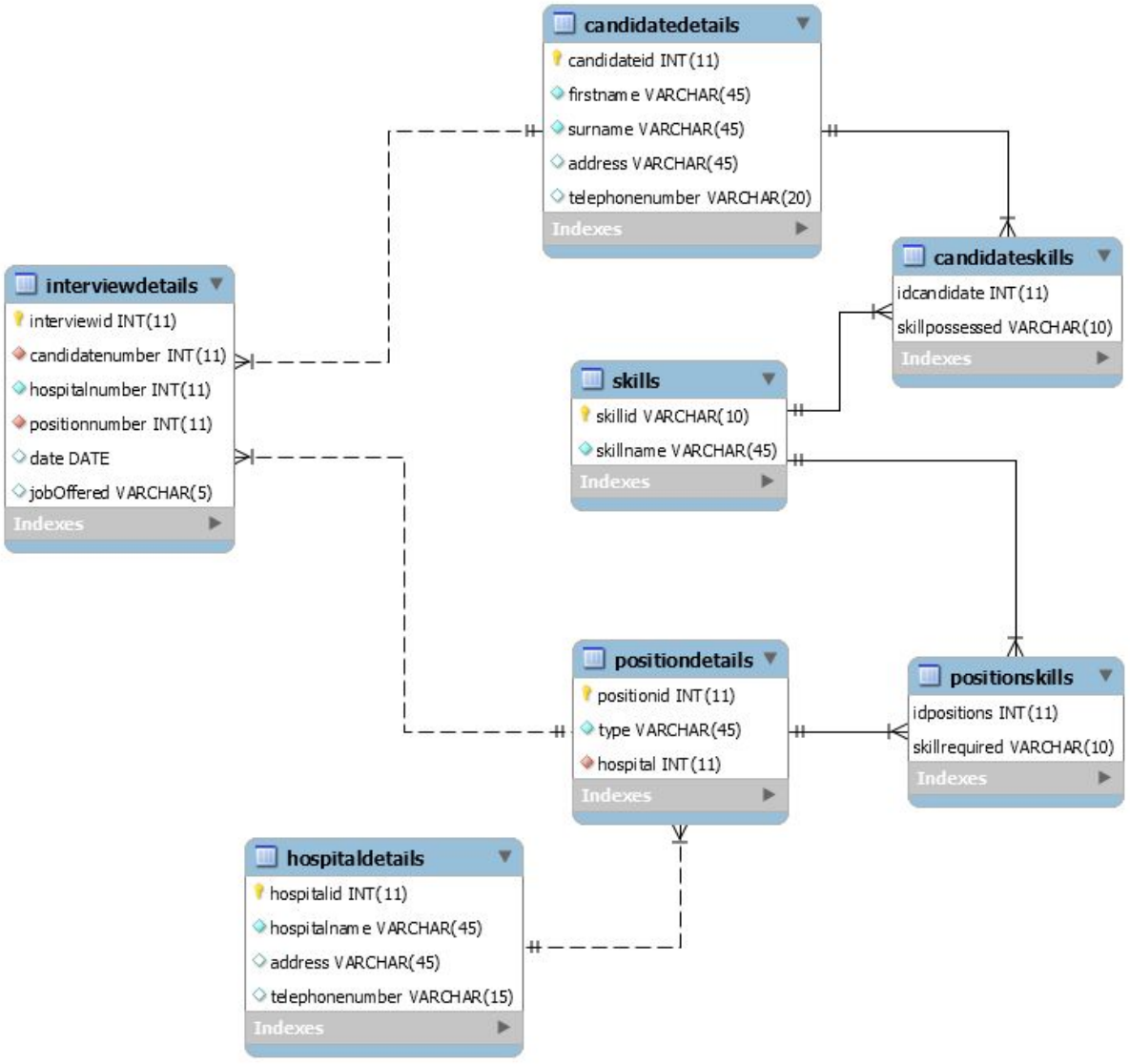
Assumptions: Job offers decisions aren’t only dependent on the skills.
I assumed the variable sizes won’t be exceeded.
Notes: I added a skills tables to make queries more efficient as in ‘positionskills’ and ‘candidateskills’ I only need to refer to the skillid, and now I can also add unused skills to the skills table and keep them there.

Glossary:

Table Name	Function	Attributes	Keys
hospitaldetails	This table has details about the hospitals in the database.	`hospitalid` int(11) `hospitalname` varchar(45) `address` varchar(45) `telephonenumber` varchar(15) ~ *used varhcar instead of int incase brackets are included.*	**Primary Key:** `hospitalid`
candidatedetails	Contains details about candidates for the positions	`candidateid` int(11) `firstname` varchar(45) `surname` varchar(45) `address` varchar(45) `telephonenumber` varchar(20)	**Primary Key:** `candidateid`
candidateskills	This table contains the skills that each candidate has.	`idcandidate` int(11) `skillpossessed` varchar(10)	**Primary Key:** `idcandidate`, `skillpossessed` ~ *as the same person may have multiple skills* **Forgein Key:** 'idcandidate' refers to 'candidateid' in candidatedetails. 'skillpossessed' refers to 'skillid' in skills.
positiondetails	This table contains the position openings	`positionid` int(11) `type` varchar(45) `hospital` int(11)	**Primary Key:** `positionid` **Forgein Key:** 'hospital' refers to 'hospitalname' in hospitaldetails.
positionskills	This table contains the candidates' skills	`idpositions` int(11) `skillrequired` varchar(10)	**Primary Key:** `idpositions`, `skillrequired` ~ *as the same position may require multiple skills* **Forgein Key:** 'idpositions' refers to 'positionid' in positiondetails. 'skillrequired' refers to 'skillid' in skills.
interviewdetails	This table contains the details of the interviews.	`interviewid` int(11) `candidatenumber` int(11) `hospitalnumber` int(11) `positionnumber` int(11) `date` date ~ *used date instead of varhcar to make it clearer.* `jobOffered` varchar(5) ~ *returns YES/NO.*	**Primary Key:** `intervewid` **Forgein Key:** 'candidatenumber' refers to 'candidateid' in candidatedetails. 'positionnumber' refers to 'positionid' in positiondetails ~ *I didn't add a forgein key to hospital id as that is in*

			positiondetails table.
skills	This table contains all the skills available	`skillid` varchar(10) `skillname` varchar(45)	**Primary Key:** `skillid`

ER diagram:



Reaction Policies:
For all:
ON UPDATE: CASCADE
ON DELETE: CASCADE
I picked cascade for all forgien keys because in case one of the forgein key attributes is deleted or updated in the parent table, we need to delete or update in the child table otherwise the data won't make sense anymore.

Operating System used:
Windows 10