**Task 1 – Step 1**

**Language Name to Language Type**

The **Language** with LanguageName ‘COBOL’ *is of* the **Language Type** with LanguageTypeName ‘procedural’

The **Language** **Type** with LanguageTypeName ‘procedural’ *contains* the **Language** with LanguageName ‘COBOL’

**PC to Room**

The **PC** with PCNr ‘pc01’ *is in* the **Room** with RoomNr ‘507’

The **Room** with RoomNr ‘507’ *contains* the **PC** with PCNr ‘pc01’

**Language Name to Worker Initials**

The **Worker** with WorkerInitials ‘PNC’ *has* expertise in **Language** with LanguageName ‘COBOL’

The **Language** with LanguageName ‘COBOL’ *is known by*the **Worker** with WorkerInitials ‘PNC’

**Worker Intials to Room**

The **Worker** with WorkerInitials ‘PNC’ *has access to* the**Room** with RoomNr ‘508’

The **Room** with RoomNr ‘508’ *is accessed by* the **Worker** with WorkerInitials ‘PNC’

**Worker Initials to PC**

The **Worker** with WorkerInitials ‘EFC’ *uses* the **PC** with PCNr ‘pc01’

The **PC** with PCNr ‘pc01’ *is used by* the **Worker** with WorkerInitials ‘EFC’

**Worker Name to Worker Initials**

The **Worker** with WorkerInitials ‘EFC’ *belongs to* the WorkerName ‘Ed Codfish’

The **Worker** with WorkerName ‘Ed Codfish’ *is known by* the WorkerInitials ‘EFC’

**Room to Language Name**

The **Room** with RoomNr ‘507’ *contains* the **Language** with LanguageName ‘Pascal’

The **Language** with LanguageName ‘Pascal’ *is in* RoomNr ‘507’

**PC to Language Name**

The **PC** with PCNr ‘pc01’ *has* the **Language** with LanguageName ‘Pascal’

The **Language** with LanguageName ‘Pascal’ *is on* the **PC** with PCNr ‘pc01’

**Language Type to Duration**

The **Language Type** with LanguageNameType ‘Declarative’ *is taught for* the **Duration** with Durationh ‘6 h’

The **Duration** with Durationh ‘6 h’ *is used for* the **Language Type** with LanguageTypeName ‘Declarative’

**Language Name to Duration**

The **Language** with LanguageName ‘Prolog’ *is taught for* a **Duration** of Durationh ‘3 h’

The **Duration** with Durationh ‘3 h’ *is used for* the **Language** with LanguageName ‘Prolog’

**Task 1 – Step 2**



**Task 1 – Step 3**

The entities WorkerName and WorkerInitials can be combined as there is an exclusive, one-to-one relationship between these entities. As an individual worker can be identified by either their Worker Initials or their full name, it is not necessary to have both. If the worker’s full name and initials are both covered under worker code, the schema can be simplified by not only removing one of these two entities, but eliminating the need to link each entity related to the worker to both the worker name and worker initials. The combination example is shown below:

i.e. The **Worker** with WorkerName ‘E. Codfish’ *has* the **Worker Name** with WorkerNameFull ‘Ed Codfish’

The **Worker** with WorkerName ‘E. Codfish’ *has* the **Worker Initials** with WorkerInitials ‘EFC’

**Arithmetic Derivations:**

One example of an arithmetically derived fact type would be the quantity of PC’s contained within a room. The quantity of PC’s within a room could be defined as PCQuantity.

1.1 – Table indicating Quantity of PC’s in a given Room Number

|  |  |  |
| --- | --- | --- |
| RoomNr | PCNr | PCQuantity |
| 507 | pc01,pc02 | 2 |
| 618 | pc03 | 1 |
| 508 | pc04, pc05 | 2 |

For example, from the information it can be seen that Room 507 has both pc01 and pc02, therefore;

The **Room** with RoomNr ‘507’ *contains* the **Quantity** with PCQuantity ‘2’

The **Quantity** with PCQuantity ‘2’ *is found in* the **Room** with RoomNr ‘507’

**Task 1 – Step 4**

* Each Worker Name is exclusive to each Worker Initial, and vice-versa
* Each unique room number will have a quantity of PC’s, but the quantity/number of PC’s is not unique to that room
* Each unique worker can have access to a room number, however the room numbers can be shared by multiple workers
* Each unique worker can use a PC, however the PC’s can be used by multiple workers
* Each unique PC number is within a particular room, however a room can contain multiple PC numbers
* Each Language Name belongs to a Language Type, however a Language Type can contain multiple Language Names
* Each Language type has a duration for which it is taught, however the duration value is not specific to that Language Type

**Task 1 – Step 5**

* Every instance of Worker Name is associated with its own Worker Initials, and vice-versa
* Every instance of RoomNr must have a PCQuantity value, even if the PC Quantity Value is 0
* Every instance of Worker Initials has access to a Room, and every room can be accessed by a worker
* Every worker has access to a PC, and every PC can be accessed by a worker
* Every PC must be contained within a Room, and from the information given every Room contains a PC
* Every Language must be installed on a PC, however not every PC is required to contain a Language
* Every Language can be located on a PC within a Room, however not every Room is required to contain a Language installed on a PC
* Every instance of a Duration belongs to a Language Type, and every Language Type is taught for a Duration
* Every instance of a Duration belongs to a Language being taught by a Worker, however not every worker teaches a Language for a Duration
* Every instance of a Duration is associated with a Language being taught, and every instance of a Language being taught involves a Duration
* Every instance of a Language belongs to a Language type, and every Language Type includes at least one Language

**Logical Derivations:**

One example of a logically derived fact would be that a PC has access to a particular Language Type, as it is not directly specified from the information given but can be logically implied. For example, pc-1 could be said to have procedural and declarative language types on it, but not functional as the languages installed, Prolog, Pascal and SQL belong to these two language types. This would be shown as:

The **PC** with PCNr ‘pc01’ contains the **Language Type** with LanguageTypeName ‘procedural’

The **PC** with PCNr ‘pc01’ contains the **Language Type** with LanguageTypeName ‘declarative’

The **Language** with LanguageTypeName ‘procedural’ *is found on* the **PC** with PCNr ‘pc01’

**Task 2 – Step 1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Project Number** | **Project Operating System** | **Team Members** | **Project Stage** | **Languages Used** |
| 101 | iOS | JM  IN  PNC | Debugging | COBOL  Prolog |
| 102 | Android | TAH  IN | Construction | Modula-2 |
| 103 | Android | EFC | Debugging | Miranda  COBOL |
| 104 | Windows | JM  TAH | Design | LISP  SQL  Miranda |

**Team to Operating System**

The **Project** with ProjectNr ‘101 *is working on/is the platform for* the **OS** with ProjectOS ‘iOS’

**Team to Workers**

The **Project** with ProjectNr ‘101 *includes/is in* the **Worker** with WorkerInitials ‘JM’

**Team to Project Stage**

The **Project** with ProjectNr ‘101’ *is in/is experienced by* the **Stage** with StageProject ‘Debugging’

**Team to Language Used**

The **Project** with ProjectNr ‘101’ *is using/is used by* the **Language** with LanguageName ‘COBOL’

**Operating System to Team Members**

The **OS** with ProjectOS ‘iOS’ *includes/is creating for* the **Worker** with WorkerInitials ‘JM’

**Operating System to Project Stage**

The **OS** with ProjectOS ‘iOS’ *is in/is occuring for* the **Stage** with StageProject ‘Debugging’

**Operating System to Languages Used**

The **OS** with ProjectOS ‘iOS’ *uses/is used for* the **Language** with LanguageName ‘COBOL’

**Worker to Languages Used to Project Number**

The **Language** with LanguageName ‘COBOL’ *is used by/using* the **Worker** with WorkerInitials ‘JM’ *working on/includes* the **Project** with ProjectNr ‘101’

**Worker to Project Stage**

The **Worker** with WorkerInitials ‘JM’ *works on/is worked on by* the **Stage** with StageProject ‘Debugging’

**Task 2 – Step 2**



**Task 2 – Step 3**

For the added entities it does not make sense to combine any of these as only the Project Number is unique, and does not have a unique one-to-one relationship with any of the other entities.

**Arithmetic Derivations:**

An arithmetically derived fact which could be made would be the number of Workers in a Project.

|  |  |  |
| --- | --- | --- |
| Project Number | Workers | Team Quantity |
| 101 | JM  IN  PNC | 3 |
| 102 | TAH  IN | 2 |
| 103 | EFC | 1 |
| 104 | NW  TAH | 2 |

Since the ‘Number of Workers’ was derived by adding the number of Worker initials associated with each Project Number, this is an arithmetically derived fact.

The **Project** with ProjectNr ‘101’ *has/is needed for* the **Team Quantity** with WorkerNr ‘3’

**Task 2 – Step 4**

Each link involving the ProjectNr is unique (from the ProjectNr side) as the project number does not repeat itself for the an entry within an entity.

**Task 2 – Step 5**

All new additions to the diagram involve mandatory role constraints as every instance of every entity belongs to another instance of another entity. Based on the information given in the above tables containing the new information, there are no exceptions.