

Base de Dados II

CTeSP – Técnico de Programação de Sistemas de Informação

**Departamento de Informática**

## **Sistema de Gestão de um Instituto de Ensino Superior**



*Figura 1 - Oracle Database.*

**Prova Realizada por: Gabriel Teixeira, João Moço e Tiago Jorge**

**Data: 13/10/2025**

## Índice

Índice	2
Índice de Figuras	3
Introdução	7
Estrutura do Projeto	8
Contexto do Negócio Modelado	8
Modelo Físico	8
Entidades ao Pormenor	10
Relacionamentos do Modelo Físico:	35
<i>Data Definition Language – DDL</i>	37
Estrutura do DDL	37
Inserção de Dados	83
Conclusão	84

## Índice de Figuras

Figura 1 - Oracle Database.	1
Figura 2 - Modelo Físico.	9
Figura 3 - Tabela "log".	11
Figura 4 - Tabela "acao_log".	11
Figura 5 - Tabela "entidade_log".	12
Figura 6 - Tabela "tipo_contacto".	12
Figura 7 - Tabela "tipo_departamento".	12
Figura 8 - Tabela "estado_inscricao".	13
Figura 9 - Tabela "estado_matricula".	13
Figura 10 - Tabela "curso".	14
Figura 11 - Tabela "tipo_curso".	14
Figura 12 - Tabela "unidade_curricular".	15
Figura 13 - Tabela "uc_curso".	15
Figura 14 - Tabela "uc_departamento".	15
Figura 15 - Tabela "uc_docente".	16
Figura 16 - Tabela "turma".	16
Figura 17 - Tabela "truno".	16
Figura 18 - Tabela "turno_aulas".	17
Figura 19 - Tabela "aula".	17
Figura 20 - Tabela "tipo_aula".	17
Figura 21 - Tabela "colaborador".	18
Figura 22 - Tabela "docente".	19
Figura 23 - Tabela "nao_docente".	19
Figura 24 - Tabela "cargo".	19
Figura 25 - Tabela "funcao_docente".	20
Figura 26 - Tabela "departamento".	20
Figura 27 - Tabela "gabinete_docente".	21
Figura 28 - Tabela "horario_colaborador".	21
Figura 29 - Tabela "estudante".	22
Figura 30 - Tabela "contacto_estudante".	22
Figura 31 - Tabela "matricula".	23
Figura 32 - Tabela "inscricao".	23
Figura 33 - Tabela "presenca".	24
Figura 34 - Tabela "nota".	24
Figura 35 - Tabela "estado_inscricao".	24
Figura 36 - Tabela "estado_matricula".	25

Figura 37 - Tabela "propina".	25
Figura 38 - Tabela "parcela_propina".	26
Figura 39 - Tabela "avaliacao".	26
Figura 40 - Tabela "tipo_avaliacao".	27
Figura 41 - Tabela "entrega".	27
Figura 42 - Tabela "estudante_entrega".	28
Figura 43 - Tabela "ficheiro_entrega".	28
Figura 44 - Tabela "recurso".	28
Figura 45 - Tabela "ficheiro_recurso".	29
Figura 46 - Tabela "polo".	30
Figura 47 - Tabela "edificio".	30
Figura 48 - Tabela "andar".	31
Figura 49 - Tabela "espaco".	31
Figura 50 - Tabela "tipo_espaco".	31
Figura 51 - Tabela "curso_area".	32
Figura 52 - Tabela "docente_aula".	32
Figura 53 - Tabela "turma_docente".	33
Figura 54 - Tabela "gabinete_docente".	33
Figura 55 - Tabela "estudante_entrega".	33
Figura 56 - Tabela "horario_colaborador".	34
Figura 57 - Tabela "uc_curso".	35
Figura 58 - Tabela "area".	35
Figura 59 - Drop nas Tabelas.	38
Figura 60 - Create e Alter Tabela "acao_log".	39
Figura 61 - Create e Alter da Tabela "andar".	39
Figura 62 - Create e Alter da Tabela "area".	40
Figura 63 - Create e Alter da Tabela "aula".	40
Figura 64 - Create e Alter da Tabela "avaliacao".	41
Figura 65 - Create e Alter da Tabela "cargo".	41
Figura 66 - Create e Alter da Tabela "colaborador".	42
Figura 67 - Create da Tabela "contacto_colaborador".	42
Figura 68 - Create e Alter da Tabela "contacto_estudante".	43
Figura 69 - Create e Alter da Tabela "curso".	43
Figura 70 - Create e Alter da Tabela "curso_area".	44
Figura 71 - Create e Alter da Tabela "departamento".	44
Figura 72 - Create e Alter da Tabela "docente".	45
Figura 73 - Create e Alter da Tabela "docente_aula".	45
Figura 74 - Create e Alter da Tabela "edificio".	46
Figura 75 - Create e Alter da Tabela "entidade_log".	46
Figura 76 - Create e Alter da Tabela "entrega".	46

Figura 77 - Create e Alter da Tabela "espaco".	47
Figura 78 - Create e Alter da Tabela "estado_inscricao".	48
Figura 79 - Create e Alter da Tabela "estado_matricula".	48
Figura 80 - Create e Alter da Tabela "estudante".	49
Figura 81 - Create e Alter da Tabela "estudante_entrega".	49
Figura 82 - Create e Alter da Tabela "ficheiro_entrega".	50
Figura 83 - Create e Alter da Tabela "ficheiro_recurso".	51
Figura 84 - Create e Alter da Tabela "funcao_docente".	51
Figura 85 - Create e Alter da Tabela "gabinete_docente".	52
Figura 86 - Create e Alter da Tabela "horario".	52
Figura 87 - Create e Alter da Tabela "horario_colaborador".	52
Figura 88 - Create e Alter da Tabela "inscricao".	53
Figura 89 - Create Alter da Tabela "log".	53
Figura 90 - Create e Alter da Tabela "matricula".	54
Figura 91 - Create e Alter da Tabela "nao_docente".	54
Figura 92 - Create e Alter da Tabela "nota".	55
Figura 93 - Create e Alter da Tabela "parcela_propina".	55
Figura 94 - Create e Alter da Tabela "polo".	56
Figura 95 - Create e Alter da Tabela "presenca".	56
Figura 96 - Create e Alter da Tabela "propina".	57
Figura 97 - Create e Alter da Tabela "recurso".	57
Figura 98 - Create e Alter da Tabela "tipo_aula".	57
Figura 99 - Create e Alter da Tabela "tipo_avaliacao".	58
Figura 100 - Create e Alter da Tabela "tipo_contacto".	58
Figura 101 - Create e Alter da Tabela "tipo_curso".	58
Figura 102 - Create e Alter da Tabela "tipo_departamento".	59
Figura 103 - Create e Alter da Tabela "tipo_espaco".	59
Figura 104 - Create e Alter da Tabela "turma".	60
Figura 105 - Create e Alter da Tabela "turma_docente".	60
Figura 106 - Create e Alter da Tabela "turno".	61
Figura 107 - Create e Alter da Tabela "turno_aulas".	61
Figura 108 - Create e Alter da Tabela "uc_curso".	62
Figura 109 - Create e Alter da Tabela "uc_departamento".	62
Figura 110 - Create e Alter da Tabela "uc_docente".	62
Figura 111 - Create e Alter da Tabela "unidade_curricular".	63
Figura 112 - Create e Alter da Tabela "utilizador".	63
Figura 113 - Alter da Tabela "andar".	64
Figura 114 - Alter da Tabela "aula".	64
Figura 115 - Alter da Tabela "avaliacao".	65
Figura 116 - Alter da Tabela "colaborador".	65

Figura 117 - Alter da Tabela "contacto_estudante".	66
Figura 118 - Alter da Tabela "curso_area".	66
Figura 119 - Alter da Tabela "curso".	66
Figura 120 - Alter da Tabela "departamento".	67
Figura 121 - Alter da Tabela "docente_aula".	67
Figura 122 - Alter da Tabela "docente".	68
Figura 123 - Alter da Tabela "edificio".	68
Figura 124 - Alter da Tabela "entrega".	69
Figura 125 - Alter da Tabela "espaco".	69
Figura 126 - Alter da Tabela "estudante_entrega".	70
Figura 127 - Alter da Tabela "estudante".	70
Figura 128 - Alter das Tabelas "ficheiro_entrega" e "ficheiro_recurso".	71
Figura 129 - Alter da Tabela "gabinete_docente".	71
Figura 130 - Alter da Tabela "horario_colaborador".	72
Figura 131 - Alter da Tabela "horario".	72
Figura 132 - Alter da Tabela "matricula".	73
Figura 133 - Alter da Tabela "inscricao".	74
Figura 134 - Alter da Tabela "log".	75
Figura 135 - Outro Alter da Tabela "matricula".	75
Figura 136 - Alter da Tabela "nao_docente".	76
Figura 137 - Alter da Tabela "nota".	76
Figura 138 - Alter da Tabela "parcela_propina".	77
Figura 139 - Alter da Tabela "presenca".	77
Figura 140 - Alter da Tabela Propina.	77
Figura 141 - Alter da Tabela "recurso".	78
Figura 142 - Alter da Tabela "turma_docente".	78
Figura 143 - Outro Alter da Tabela "espaco".	79
Figura 144 - Alter da Tabela "turma".	79
Figura 145 - Alter da Tabela "uc_curso".	80
Figura 146 - Alter da Tabela "uc_departamento".	81
Figura 147 - Alter da Tabela "uc_docente".	82

## Introdução

O presente documento refere-se à primeira fase do trabalho final da disciplina, que vem sendo realizado ao longo das últimas semanas, no âmbito da unidade curricular de Bases de Dados II, lecionada pelo professor Luís Travassos.

O mesmo visa abordar de forma abrangente, mas também dar continuidade, aos conceitos lecionados na disciplina de Bases de Dados I no último semestre

Isto inclui:

- Diagrama Entidade-Relacionamento (ER) do esquema proposto;
- *Script SQL* com os comandos *CREATE TABLE* e *ALTER TABLE*;
- *Script SQL* com os comandos *INSERT* para popular as tabelas;
- Breve descrição do contexto do negócio modelado;

Todos os tópicos referidos acima, serão abordados no decorrer deste documento.

## Estrutura do Projeto

O projeto foi desenvolvido com recurso às ferramentas: *Data Modeler* que foi usada para modelar e produzir o diagrama físico, bem como o DDL que por sua vez foi exportado para o *SQL Developer*, para fazer a parte da inserção de dados no projeto.

Esta última etapa (inserção de dados), pôde contar com a ajuda do *Mockaroo*, que é uma ferramenta online que permite gerar dados falsos (simulados) de forma altamente personalizada.

## Contexto do Negócio Modelado

O nosso grupo escolheu o tema da universidade porque este tipo de sistema representa um cenário realista, complexo e altamente relevante no contexto atual. Universidades lidam diariamente com grandes volumes de dados relacionados a estudantes, professores, cursos, disciplinas, matrículas, avaliações, etc.

Ademais, é um tema com o qual todos temos familiaridade, o que facilita a compreensão dos requisitos e a definição das entidades e atributos necessários. Também consideramos que esta escolha proporciona oportunidades futuras de inclusão de outros conceitos aprendido na disciplina de Base de Dados II, como os *triggers*, funções, cursores e procedimentos.

Por fim, acreditamos que é uma excelente forma de desenvolver habilidades que poderão ser aplicadas em diversos contextos profissionais, dado que a gestão eficiente dos dados é uma necessidade presente em várias áreas.

## Modelo Físico

Após perceber qual era a tarefa imposta para realizar na primeira fase, e de terem sido identificadas, tanto as entidades, como os atributos de cada entidade e ainda os relacionamentos que as entidades teriam entre elas, chegava a hora de traduzir todos esses dados/informação para o *Data Modeler*.

O *Data Modeler*, serve para criar, visualizar, modificar e gerir modelos de dados, desde a sua fase mais conceitual até ao modelo físico. Por isso foi produzido o modelo físico, que no fundo acaba por ser uma representação mais detalhada e refinada do diagrama de entidade-relacionamento, o seu objetivo foi representar as entidades como tabelas, os atributos como colunas, e os relacionamentos como chaves estrangeiras.



De referir ainda que, este modelo segue os princípios da normalização (ou seja, elimina redundâncias e melhora a integridade dos dados).

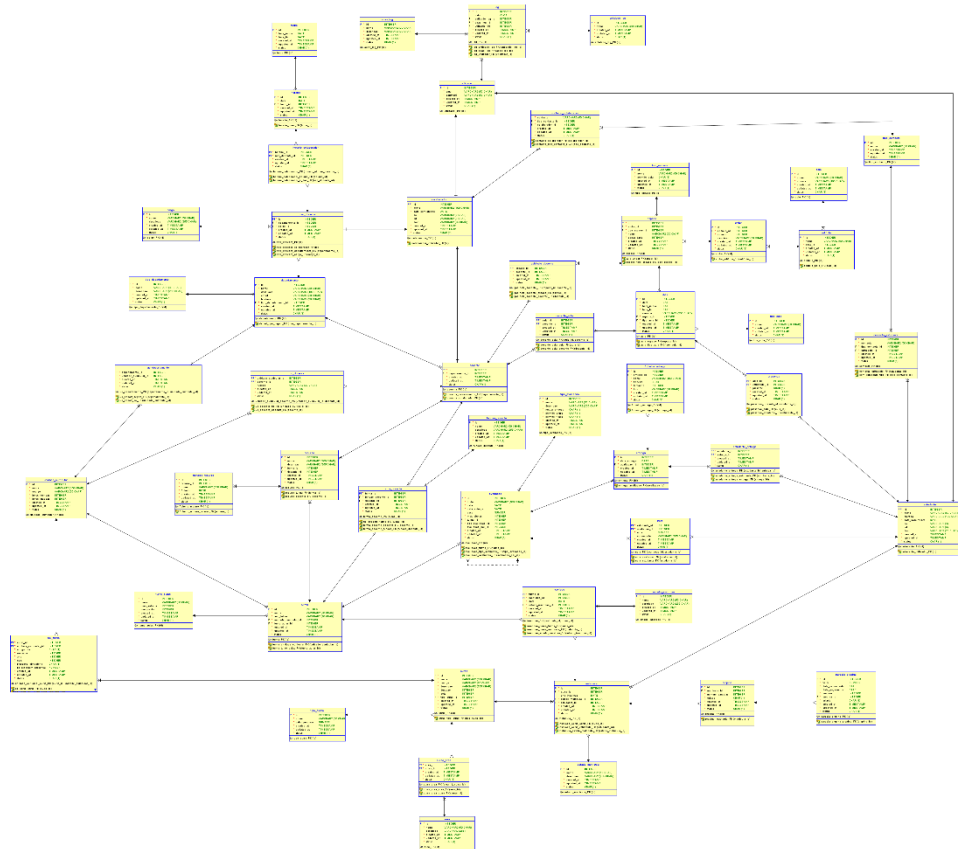


Figura 2 - Modelo Físico.

Este modelo físico comporta entidades separadas por nove tipos de entidades, sendo elas:

1. **Entidades de Utilidade Geral:** Isto é, tabelas genéricas de apoio à infraestrutura e de apoio ao sistema da universidade;
2. **Entidades Académicas Principais:** Estas representam os conceitos centrais do ensino, ou seja, tem uma estrutura mais pedagógica e curricular;
3. **Entidades de Pessoal:** Dividas entre docentes e não docentes, basicamente representam a gestão de recurso humanos da universidade;
4. **Entidades de Estudantes:** Mais focadas em dados pessoais e percurso académico;

5. **Entidades Financeiras:** Relacionadas a propinas e pagamentos, comportam a gestão financeira;
6. **Entidades de Avaliação e Recursos Didáticos:** Neste caso, controlam as avaliações e materiais de apoio;
7. **Entidades Físicas/Infraestrutura:** Isto é, relacionadas com espaços e edifícios, no fundo a infraestrutura física;
8. **Entidades de Associação/Relação:** Servem mais para ligar várias entidades principais (tabelas de junção), citadas acima;
9. **Entidades de Meta dados/Áreas:** Tendem a falar mais da área científica do curso;

Resumindo por tipo, existem sete entidades relativas à utilidade geral (sistema, tipos, logs), existem nove entidades académicas, oito entidades de pessoal, sete entidades de estudantes, duas entidades com foco financeiro da universidade, seis entidades sobre avaliações e recursos didáticos, cinco entidades físicas/infraestrutura, oito entidades de associação/relação e por fim uma única entidade relativa a área científica do curso. Totalizando 53 entidades, incluindo normalização.

### Legenda Modelo Físico

A título de elucidação e antes de fechar em definitivo este capítulo relativo ao modelo físico, é necessário fazer a legenda de outros elementos contidos nesse preciso modelo.

- **Asteriscos Antes de Atributos** – Significa que são atributos obrigatórios no quesito de serem preenchidos;
- **P** – A sigla P é relativa a *Primary Key*;
- **F** – A letra F, indica uma *Foreign Key*;
- **N:1** – Relacionamento um para muitos;

### Entidades ao Pormenor

Este tópico visa explicar ao pormenor cada uma das cinquenta e três entidades (agrupando-as por género naturalmente), isto porque o modelo físico é enorme.

### Entidades de Utilidade Geral

Começando por entidades de utilidade geral, como já referido acima, existem sete e apesar de parecer complexo, é fácil subdividir.

### Tabela “log”:

Esta tabela, é uma tabela de utilidade geral, existe para suprir a necessidade de registar e armazenar *logs*.

log	
P * id	INTEGER
data	CLOB
F entidade_log_id	INTEGER
F * acao_log_id	INTEGER
F * utilizador_id	INTEGER
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
log_PK (id)	
log_entidade_log_FK (entidade_log_id)	
log_acao_log_FK (acao_log_id)	
log_utilizador_FK (utilizador_id)	

Figura 3 - Tabela "log".

### Tabela “acao\_log” :

É uma tabela de utilidade geral, isto é, ela acaba por guardar ações registadas no *log* (como por exemplo, a criação, atualização e remoção de dados).

acao_log	
P * id	INTEGER
* nome	VARCHAR2 (20 CHAR)
* descricao	VARCHAR2 (20 CHAR)
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
acao_log_PK (id)	

Figura 4 - Tabela "acao\_log".

### Tabela “entidade\_log”:

Relativamente a esta tabela, é uma tabela de utilidade geral, como tal, surgiu a necessidade de haverem outras entidades que podiam gerar *logs*, daí a criação da mesma.

entidade_log		
P	* id	INTEGER
	* nome	VARCHAR2 (50 CHAR)
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
entidade_log_PK (id)		

Figura 5 - Tabela "entidade\_log".

### Tabela "tipo\_contacto":

Esta tabela, já não tem como finalidade guardar *logs*, mas sim contactos, daí ser na mesma uma tabela de utilidade geral no sistema da universidade.

Ela guarda contactos, tipo *e-mail*, telefone, etc.

tipo_contacto		
P	* id	INTEGER
	* nome	VARCHAR2 (50 CHAR)
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
tipo_contacto_PK (id)		

Figura 6 - Tabela "tipo\_contacto".

### Tabela "tipo\_departamento":

Esta tabela, serve para guardar o tipo ou categoria de um departamento da universidade e como tal, mantém na categoria de entidades de utilidade geral no sistema.

tipo_departamento		
P	* id	INTEGER
	* nome	VARCHAR2 (30 CHAR)
	* descricao	VARCHAR2 (255 CHAR)
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
tipo_departamento_PK (id)		

Figura 7 - Tabela "tipo\_departamento".

### Tabela "estado\_inscricao":

Esta tabela, mantém-se na mesma categoria que as anteriores, o seu propósito é justamente armazenar o estado de inscrição (se está ativa, se está anulada, etc).

estado_inscricao		
P	* id	INTEGER
	* nome	VARCHAR2 (50 CHAR)
	* descricao	VARCHAR2 (255 CHAR)
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
estado_inscricao_PK (id)		

Figura 8 - Tabela "estado\_inscricao".

### Tabela “estado\_matricula”:

Já esta tabela, tem o mesmo propósito da anterior, mas armazena o estado da matrícula (se a matrícula está ativa, se está anulada, se está suspensa, concluída, etc).

estado_matricula		
P	* id	INTEGER
	* nome	VARCHAR2 (50 CHAR)
	* descricao	VARCHAR2 (255 CHAR)
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
estado_matricula_PK (id)		

Figura 9 - Tabela "estado\_matricula".

### Entidades Académicas Principais:

Neste tópico, fala-se das entidades académicas, que como já referido são uma grande parte do sistema, contando com nove entidades.

### Tabela “curso”:

A tabela “curso”, vai armazenar todos os cursos daquela unidade de ensino superior.

curso		
P	* id	INTEGER
	* nome	VARCHAR2 (100 CHAR)
	* codigo	VARCHAR2 (20 CHAR)
	* descricao	VARCHAR2 (255 CHAR)
	* duracao	INTEGER
	* ects	INTEGER
F	* tipo_curso_id	INTEGER
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
curso_PK (id)		
curso_tipo_curso_FK (tipo_curso_id)		

Figura 10 - Tabela "curso"

### Tabela “tipo\_curso”:

Esta tabela é na verdade uma sub-tabela da anterior (curso), que permite armazenar não o curso, mas sim o tipo de curso (CTeSP, licenciatura, mestrado, etc...).

tipo_curso	
P * id	INTEGER
* nome	VARCHAR2 (50 CHAR)
* valor_propinas	NUMBER
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
tipo_curso_PK (id)	

Figura 11 - Tabela "tipo\_curso".

### Tabela “unidade\_curricular”:

Esta tabela diz respeito precisamente à unidade curricular/disciplina.

unidade_curricular	
P * id	INTEGER
* nome	VARCHAR2 (100 CHAR)
* codigo	VARCHAR2 (20 CHAR)
* horas_teoricas	INTEGER
* horas_praticas	INTEGER
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
unidade_curricular_PK (id)	

Figura 12 - Tabela "unidade\_curricular".

### Tabela “uc\_curso”:

Esta tabela é na verdade uma associação entre a tabela “curso” e a tabela “unidade\_curricular”.

uc_curso	
PF* curso_id	INTEGER
PF* unidade_curricular_id	INTEGER
* obrigatoria	CHAR (1)
* semestre	INTEGER
* ano	INTEGER
* ects	INTEGER
* presenca_obrigatoria	CHAR (1)
* percentagem_presenca	NUMBER
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
unidade_curricular_curso_PK (curso_id, unidade_curricular_id)	
uc_curso_curso_FK (curso_id)	

Figura 13 - Tabela "uc\_curso".

### Tabela "uc\_departamento":

Esta tabela tem o mesmo propósito da anterior, mas desta vez é uma associação entre a tabela "unidade\_curricular" e a tabela "departamento".

uc_departamento	
PF* departamento_id	INTEGER
PF* unidade_curricular_id	INTEGER
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
uc_departamento_PK (departamento_id, unidade_curricular_id)	
uc_depart_depart_FK (departamento_id)	
uc_depart_uc_FK (unidade_curricular_id)	

Figura 14 - Tabela "uc\_departamento".

### Tabela "uc\_docente":

Mais uma tabela de associação, mas desta vez entre "unidade curricular" e a tabela "docente".

uc_docente	
PF* unidade_curricular_id	INTEGER
PF* docente_id	INTEGER
* funcao	VARCHAR2 (30 CHAR)
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
unidade_curricular_docente_PK (unidade_curricular_id, docente_id)	
uc_docente_uc_FK (unidade_curricular_id)	
uc_docente_docente_FK (docente_id)	

Figura 15 - Tabela "uc\_docente".

### Tabela "turma":

A tabela turma, permite guardar dados relativos à turma, relativamente a determinada unidade curricular.

turma		
P	* id	INTEGER
	* nome	VARCHAR2 (20 CHAR)
	* ano_letivo	VARCHAR2 (10 CHAR)
F	* unidade_curricular_id	INTEGER
F	* turno_aulas_id	INTEGER
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
turma_PK (id)		
turma_unidade_curricular_FK (unidade_curricular_id)		
turma_turno_aulas_FK (turno_aulas_id)		

Figura 16 - Tabela "turma".

### Tabela “turno”:

Por sua vez, a tabela “turno”, diz respeito ao tipo de horário que ocorrem as aulas (por exemplo, pós laboral, horário integral, etc).

turno		
P	* id	INTEGER
	* hora_inicio	DATE
	* hora_fim	DATE
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
turno_PK (id)		

Figura 17 - Tabela "truno".

### Tabela “turno\_aulas”:

Embora com um nome muito idêntico à tabela anterior, esta tabela trata de lidar com dados relativos ao grupo de aulas por turno.

turno_aulas		
P	* id	INTEGER
	* nome	VARCHAR2 (50 CHAR)
	hora_entrada	INTEGER
	hora_saida	INTEGER
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
turno_aulas_PK (id)		

Figura 18 - Tabela "turno\_aulas".

### Tabela “aula”:

Mais uma das entidades principais deste sistema de gestão de universidade, esta tabela visa armazenar dados relativos à aula em específico.



aula	
P	* id INTEGER
	* data DATE
	* hora_inicio DATE
	* hora_fim DATE
	sumario VARCHAR2 (255 CHAR)
F	* espaco_id INTEGER
F	* tipo_aula_id INTEGER
	* created_at TIMESTAMP
	* updated_at TIMESTAMP
	* status CHAR (1)
aula_PK (id)	
aula_espaco_FK (espaco_id)	
aula_tipo_aula_FK (tipo_aula_id)	

Figura 19 - Tabela "aula".

### Tabela “tipo\_aula”:

Mais uma sub-tabela, mas desta vez da tabela “aula”. Nesta tabela guardam-se dados relativos a se a aula é teórica, prática, laboratorial, etc...

tipo_aula	
P	* id INTEGER
	* nome VARCHAR2 (50 CHAR)
	* created_at TIMESTAMP
	* updated_at TIMESTAMP
	* status CHAR (1)
tipo_aula_PK (id)	

Figura 20 - Tabela "tipo\_aula".

### Entidades de Pessoal

Continuando a pormenorizar as diversas entidades desde sistema de gestão de uma universidade, chegou-se às entidades relacionadas com pessoal da universidade.

### Tabela “colaborador”:

Esta tabela, é fácil de decodificar, isto porque como o próprio nome diz, a mesma armazena dados relativos a pessoas que colaboram com a instituição (desde professores, funcionários, etc...).

colaborador	
PF * id	INTEGER
* nome	VARCHAR2 (100 CHAR)
* data_contratacao	DATE
cc	VARCHAR2 (8 CHAR)
nif	VARCHAR2 (9 CHAR)
iban	VARCHAR2 (25 CHAR)
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
colaborador_PK (id)	
colaborador_utilizador_FK (id)	

Figura 21 - Tabela "colaborador".

### Tabela “docente”:

Acima armazenavam-se dados relativos aos colaboradores e agora nesta tabela armazenam-se dados relativos aos docentes.

docente	
PF * id	INTEGER
F * departamento_id	INTEGER
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
docente_PK (id)	
docente_departamento_FK (departamento_id)	
docente_colaborador_FK (id)	

Figura 22 - Tabela "docente".

### Tabela “não\_docente”:

Esta tabela diz respeito mais propriamente a pessoal colaborador, mas não docente.

nao_docente	
PF *	id INTEGER
F *	departamento_id INTEGER
F *	cargo_id INTEGER
	created_at TIMESTAMP
	updated_at TIMESTAMP
	status CHAR (1)
nao_docente_PK (id)	
nao_docente_colaborador_FK (id)	
nao_docente_departamento_FK (departamento_id)	
nao_docente_cargo_FK (cargo_id)	

Figura 23 - Tabela "nao\_docente".

### Tabela “cargo”:

Neste caso a tabela “cargo” armazena dados relativos ao cargo do pessoal colaborador, mas não docente (funcionário, administrador, presidente, etc...).

cargo	
P *	id INTEGER
	nome VARCHAR2 (50 CHAR)
	descricao VARCHAR2 (255 CHAR)
	created_at TIMESTAMP
	updated_at TIMESTAMP
	status CHAR (1)
cargo_PK (id)	

Figura 24 - Tabela "cargo".

### Tabela “funcao\_docente”:

Se anteriormente se falou de cargos para pessoal não docente, esta tabela vai guardar funções/cargo de docentes (se é professor assistente, se é catedrático, etc...).

funcao_docente	
P *	id INTEGER
	nome VARCHAR2 (50 CHAR)
	descricao VARCHAR2 (255 CHAR)
	created_at TIMESTAMP
	updated_at TIMESTAMP
	status CHAR (1)
funcao_docente_PK (id)	

Figura 25 - Tabela "funcao\_docente".

### Tabela “departamento”:

Esta entidade também se pode classificar com uma entidade principal, isto porque a mesma guardar dados relativos ao departamento académico.

departamento		
P	* id	INTEGER
	* nome	VARCHAR2 (30 CHAR)
	* descricao	VARCHAR2 (100 CHAR)
	* email	VARCHAR2 (50 CHAR)
	* telefone	VARCHAR2 (20 CHAR)
F	* tipo_departamento_id	INTEGER
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
departamento_PK (id)		
depart_tipo_depart_FK (tipo_departamento_id)		

Figura 26 - Tabela "departamento".

### Tabela “gabinete\_docente”:

Esta tabela, serve para associar a tabela “docente” e o seu gabinete (tabela “espaco”).

gabinete_docente		
PF	* espaco_id	INTEGER
PF	* docente_id	INTEGER
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
gabinete_docente_PK (espaco_id, docente_id)		
gabinete_docente_espaco_FK (espaco_id)		
gabinete_docente_docente_FK (docente_id)		

Figura 27 - Tabela "gabinete\_docente".

### Tabela “horário\_colaborador”:

Neste documento já se sabe da existência de uma entidade relativa aos colaboradores, então esta tabela serve para ligar a tabela “horario” e a tabela “nao\_docente”.

horario_colaborador	
PF*	horario_id INTEGER
PF*	nao_docente_id INTEGER
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
horario_colaborador_PK(horario_id, nao_docente_id)	
horario_colaborador_horario_FK(horario_id)	
horario_colaborador_n_doc_FK(nao_docente_id)	

Figura 28 - Tabela "horario\_colaborador".

## Entidades de Estudantes

Neste tópico falar-se-á de entidades relativas a estudantes da universidade retratada ao longo deste documento.

### Tabela “estudante”:

Esta entidade é considerada uma entidade principal deste sistema, por isso mesmo esta entidade armazena os dados pessoais do estudante.

estudante		
PF *	id	INTEGER
*	nome	VARCHAR2 (50 CHAR)
	morada	VARCHAR2 (100 CHAR)
*	data_nascimento	DATE
	cc	VARCHAR2 (8)
	nif	VARCHAR2 (9)
	iban	VARCHAR2 (25 CHAR)
*	created_at	TIMESTAMP
*	updated_at	TIMESTAMP
*	status	CHAR (1)
estudante_PK (id)		
estudante_utilizador_FK (id)		

Figura 29 - Tabela "estudante".

### Tabela “contacto\_estudante”:

No caso desta tabela, ela nada mais faz do que armazenar os contactos dos estudantes.

contacto_estudante		
P *	id	INTEGER
*	contacto	VARCHAR2 (50 CHAR)
F *	tipo_contacto_id	INTEGER
F *	estudante_id	INTEGER
	created_at	TIMESTAMP
	updated_at	TIMESTAMP
	status	CHAR (1)
contacto_PK (id)		
contacto_estudante_FK (estudante_id)		
cont_estudante_tipo_FK (tipo_contacto_id)		

Figura 30 - Tabela "contacto\_estudante".

### Tabela “matricula”:

Esta entidade como é natural, irá reter dados que dizem respeito às matrículas dos diversos estudantes num curso.

matricula		
P *	id	INTEGER
F *	curso_id	INTEGER
	ano_inscricao	DATE
F *	estado_matricula_id	INTEGER
F *	estudante_id	INTEGER
	created_at	TIMESTAMP
	updated_at	TIMESTAMP
	status	CHAR (1)
matricula_PK (id)		
	inscricao_curso_curso_FK (curso_id)	
	inscricao_curso_estudante_FK (estudante_id)	
	matricula_estado_matricula_FK (estado_matricula_id)	

Figura 31 - Tabela "matricula".

### Tabela “inscricao”:

Esta tabela, também é de fácil percepção, uma vez que ira guardar os dados de inscrições em unidades curriculares ou turmas.

inscricao		
PF *	turma_id	INTEGER
PF *	matricula_id	INTEGER
	data	DATE
F *	estado_inscricao_id	INTEGER
	created_at	TIMESTAMP
	updated_at	TIMESTAMP
	status	CHAR (1)
inscricao_PK (matricula_id, turma_id)		
	inscricao_turma_turma_FK (turma_id)	
	inscricao_turma_matricula_FK (matricula_id)	
	inscricao_estado_inscricao_FK (estado_inscricao_id)	

Figura 32 - Tabela "inscricao".

### Tabela “presença”:

Esta entidade armazena precisamente o registo de presenças em aulas.

presenca	
PF*	aula_id INTEGER
PF*	estudante_id INTEGER
*	presente CHAR (1)
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
presenca_PK (aula_id, estudante_id)	
presenca_aula_FK (aula_id)	
presenca_estudante_FK (estudante_id)	

Figura 33 - Tabela "presenca".

### Tabela “nota”:

Esta tabela diz respeito às notas dos alunos e como tal, armazena as mesmas, bem como possíveis comentários de avaliação.

nota	
PF*	estudante_id INTEGER
PF*	avaliacao_id INTEGER
*	nota NUMBER
*	comentario VARCHAR2 (255 CHAR)
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
nota_PK (avaliacao_id, estudante_id)	
nota_avaliacao_FK (avaliacao_id)	
nota_estudante_FK (estudante_id)	

Figura 34 - Tabela "nota".

### Tabela “estado\_inscricao”:

Acima foi reportada uma entidade denominada por “inscricao”, como tal esta tabela irá guardar o estado da mesma.

estado_inscricao	
P*	id INTEGER
*	nome VARCHAR2 (50 CHAR)
*	descricao VARCHAR2 (255 CHAR)
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
estado_inscricao_PK (id)	

Figura 35 - Tabela "estado\_inscricao".

### Tabela “estado\_matricula”:



À semelhança da tabela anterior, neste documento já foi falado de uma entidade denominada de “matricula”, como tal esta entidade armazena o estado dessa mesma matrícula.

estado_matricula	
P * id	INTEGER
* nome	VARCHAR2 (50 CHAR)
descricao	VARCHAR2 (255 CHAR)
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
estado_matricula_PK (id)	

Figura 36 - Tabela "estado\_matricula".

### Entidades Financeiras:

Prosseguindo na desmistificação das diversas entidades, chegou-se ao ponto de se falar das entidades financeiras.

### Tabela “propina”:

Mais um caso de uma entidade principal deste sistema. Esta tabela armazena precisamente a propina associada à matrícula.

propina	
P * id	INTEGER
F * matricula_id	INTEGER
* numero_parcelas	INTEGER
* estado	CHAR (1)
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
propina_PK (id)	
propina_matricula_FK (matricula_id)	

Figura 37 - Tabela "propina".

### Tabela “parcela\_propina”:

Esta tabela representa na verdade uma sub-tabela da tabela “propina”. Posto isto, o seu propósito é armazenar os dados relativos às parcelas de pagamento de propinas.

parcela_propina	
P *	id INTEGER
*	valor NUMBER
*	data_vencimento DATE
	data_pagamento DATE
*	numero INTEGER
F *	propina_id INTEGER
*	estado CHAR (1)
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
parcela_propina_PK (id)	
parcela_propina_propina_FK (propina_id)	

Figura 38 - Tabela “parcela\_propina”.

#### Entidades de Avaliação e Recursos Didáticos:

Quase a acabar, mas ainda importa falar de sete entidades relacionadas ao tópico no título descrito.

#### Tabela “avaliacao”:

Mais um caso de uma entidade principal, esta tabela armazenará os dados relativos às avaliações dos alunos.

avaliacao	
P *	id INTEGER
*	titulo VARCHAR2 (100 CHAR)
*	data DATE
*	data_entrega DATE
	peso NUMBER
*	max_alunos INTEGER
F *	turma_id INTEGER
F *	tipo_avaliacao_id INTEGER
F *	avaliacao_pai_id INTEGER
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
avaliacao_PK (id)	
avaliacao_turma_FK (turma_id)	
avaliacao_tipo_avaliacao_FK (tipo_avaliacao_id)	
avaliacao_avaliacao_FK (avaliacao_pai_id)	

Figura 39 - Tabela “avaliacao”.

#### Tabela “tipo\_avaliacao”:

Mais uma sub-tabela, desta vez é uma sub-tabela da entidade “avaliacao” e como tal irá guardar dados relativos ao tipo de avaliação, isto é, se é exame, frequência, trabalhos, etc...

tipo_avaliacao	
P * id	INTEGER
* nome	VARCHAR2 (50 CHAR)
* descricao	VARCHAR2 (255 CHAR)
* requer_entrega	CHAR (1)
* permite_grupo	CHAR (1)
* permite_filhos	CHAR (1)
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
tipo_avaliacao_PK (id)	

Figura 40 - Tabela "tipo\_avaliacao".

### Tabela “entrega”:

Esta tabela é uma tabela de associação e relativa à entidade “avaliacao”, como tal guarda dados relativos às entregas para avaliações.

entrega	
P * id	INTEGER
* data_entrega	DATE
F * avaliacao_id	INTEGER
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
entrega_PK (id)	
entrega_avaliacao_FK (avaliacao_id)	

Figura 41 - Tabela "entrega".

### Tabela “estudante\_entrega”:

Falando da tabela “estudante\_entrega”, tal como o nome revela, faz a associação entre a entidade “estudante” e “entrega”.

estudante_entrega	
PF * estudante_id	INTEGER
PF * entrega_id	INTEGER
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
estudante_entrega_PK (estudante_id, entrega_id)	
estudante_entrega_estudante_FK (estudante_id)	
estudante_entrega_entrega_FK (entrega_id)	

Figura 42 - Tabela "estudante\_entrega".

### Tabela “ficheiro\_entrega”:

Para complementar ainda mais a parte de entrega de materiais para avaliação, nasce a necessidade de criar esta tabela.

ficheiro_entrega		
P	* id	INTEGER
F	* entrega_id	INTEGER
	* nome	VARCHAR2 (100 CHAR)
	* ficheiro	BLOB
	* tamanho	INTEGER
	* tipo	VARCHAR2 (20 CHAR)
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
ficheiro_entrega_PK (id)		
ficheiro_entrega_FK (entrega_id)		

Figura 43 - Tabela "ficheiro\_entrega".

### Tabela “recurso”:

Para os alunos que infelizmente têm azar e têm de ir a recurso, surge esta tabela que visa armazenar precisamente os dados do recurso.

recurso		
P	* id	INTEGER
	* nome	VARCHAR2 (100 CHAR)
	* descricao	VARCHAR2 (255 CHAR)
F	* turma_id	INTEGER
F	* docente_id	INTEGER
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
recurso_PK (id)		
recurso_turma_FK (turma_id)		
recurso_docente_FK (docente_id)		

Figura 44 - Tabela "recurso".

### Tabela “ficheiro\_recurso”:

Uma vez mais, houve necessidade de complementar a entidade recurso e como tal surgiu a tabela “ficheiro\_recurso”, que guarda os ficheiros de recurso, como PDF’s por exemplo.

ficheiro_recurso		
P	* id	INTEGER
F	* recurso_id	INTEGER
	* nome	VARCHAR2 (100 CHAR)
	* ficheiro	BLOB
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
ficheiro_recurso_PK (id)		
ficheiro_recurso_recurso_FK (recurso_id)		

Figura 45 - Tabela "ficheiro\_recurso".

### Entidades Físicas/Infraestrutura:

Faltando apenas três tópicos para terminar este capítulo de observação ao pormenor das diferentes entidades do sistema de gestão da universidade, falar-se-á agora de entidades relativas à infraestrutura da universidade.

### Tabela “polo”:

Começando pela entidade principal deste tópico, é razoavelmente entendível que esta entidade irá reter todos os dados relativos ao edifício, como quem o polo/campus universitário.

polo		
P	* id	INTEGER
	* nome	VARCHAR2 (50 CHAR)
	* morada	VARCHAR2 (100 CHAR)
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
polo_PK (id)		

Figura 46 - Tabela "polo".

### Tabela “edifício”:

Se anteriormente se guardaram dados na entidade “polo”, na entidade “edifício” adensa-se ainda mais o fator de armazenamento de dados do edifício (instalações da universidade”.

edifício		
P	* id	INTEGER
	* nome	VARCHAR2 (50 CHAR)
F	* polo_id	INTEGER
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
edifício_PK (id)		
edifício_polo_FK (polo_id)		

Figura 47 - Tabela "edifício".

### Tabela “andar”:

Já deu para entender que existe uma entidade “polo” e uma entidade “edifício”, mas os dados relativos ao andar do edifício também são armazenados.

andar		
P	* id	INTEGER
F	* edificio_id	INTEGER
	* numero	INTEGER
	* descricao	VARCHAR2 (50 CHAR)
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
andar_PK (id)		
andar_edificio_FK (edificio_id)		

Figura 48 - Tabela "andar".

### Tabela “espaco”:

Na tabela “espaco” é possível guardar dados relativos ao espaço físico, isto inclui por exemplo, salas, laboratórios, gabinetes de professor/atendimento.

espaco	
P * id	INTEGER
F * andar_id	INTEGER
F * tipo_espaco_id	INTEGER
* nome	VARCHAR2 (50 CHAR)
* capacidade	INTEGER
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
=> espaco_PK (id)	
=> sala_andar_FK (andar_id)	
=> espaco_tipo_espaco_FK (tipo_espaco_id)	

Figura 49 - Tabela "espaco".

### Tabela "tipo\_espaco":

Por último e ainda relativamente a entidades que visam guardar dados da estrutura física, há uma tabela que guarda o tipo de espaço, que é uma sub-tabela da tabela "espaco" e vai essencialmente guardar a disponibilidade das salas para permitir aulas ou não.

tipo_espaco	
P * id	INTEGER
* nome	VARCHAR2 (50 CHAR)
* permite_aula	CHAR (1)
* created_at	TIMESTAMP
* updated_at	TIMESTAMP
* status	CHAR (1)
=> tipo_espaco_PK (id)	

Figura 50 - Tabela "tipo\_espaco".

### Entidades de Associação e Relação:

Como já referido anteriormente no presente documento, existem entidades mais focadas em guardar certas relações entre entidades principais e é precisamente isso que este tópico visa descrever.

Relembrando que algumas destas tabelas, já foram faladas anteriormente, mas por uma questão de fácil percepção e explicação, é preferível descrevê-las.

### Tabela "curso\_area":

Trata-se de uma tabela que serve para guardar dados que são fruto da relação da tabela "curso" e da tabela "área" (esta última, ainda não foi descrita).

curso_area	
PF*	area_id INTEGER
PF*	curso_id INTEGER
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
curso_area_PK (area_id, curso_id)	
curso_area_area_FK (area_id)	
curso_area_curso_FK (curso_id)	

Figura 51 - Tabela "curso\_area".

### Tabela "docente\_aula":

Espelhando a tabela anterior, esta tabela surgiu da necessidade de guardar dados entre a relação da entidade "docente" e a entidade "aula".

docente_aula	
PF*	aula_id INTEGER
PF*	docente_id INTEGER
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
docente_aula_PK (aula_id, docente_id)	
docente_aula_aula_FK (aula_id)	
docente_aula_docente_FK (docente_id)	

Figura 52 - Tabela "docente\_aula".

### Tabela "turma\_docente":

Seguindo o mesmo pensamento do tópico anterior, esta tabela descreve a relação entre a entidade principal "turma" e a entidade principal "docente".

turma_docente	
PF*	turma_id INTEGER
F*	funcao_docente_id INTEGER
F*	docente_id INTEGER
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
turma_docente_PK (turma_id)	
uc_docente_turma_FK (turma_id)	
turma_docente_docente_FK (docente_id)	
turma_docente_funcao_FK (funcao_docente_id)	

Figura 53 - Tabela "turma\_docente".



### Tabela “gabinete\_docente”:

Continuando, mais uma tabela para guardar dados furtuitos das entidades “docente” e “espaco”, no caso cada professor tem o seu gabinete.

gabinete_docente	
PF*	espaco_id INTEGER
PF*	docente_id INTEGER
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
gabinete_docente_PK (espaco_id, docente_id)	
gabinete_docente_espaco_FK (espaco_id)	
gabinete_docente_docente_FK (docente_id)	

Figura 54 - Tabela "gabinete\_docente".

### Tabela “estudante\_entrega”:

Esta tabela é o resultado da relação entre as entidades principais “estudante” e “entrega”.

estudante_entrega	
PF*	estudante_id INTEGER
PF*	entrega_id INTEGER
*	created_at TIMESTAMP
*	updated_at TIMESTAMP
*	status CHAR (1)
estudante_entrega_PK (estudante_id, entrega_id)	
estudante_entrega_estudante_FK (estudante_id)	
estudante_entrega_entrega_FK (entrega_id)	

Figura 55 - Tabela "estudante\_entrega".

### Tabela “horário\_colaborador”:

Uma associação entre a entidade “horario” e a entidade “não\_docente”, no caso um não docente é um colaborador da universidade.

horario_colaborador		
PF*	horario_id	INTEGER
PF*	nao_docente_id	INTEGER
*	created_at	TIMESTAMP
*	updated_at	TIMESTAMP
*	status	CHAR (1)
horario_colaborador_PK (horario_id, nao_docente_id)		
horario_colaborador_horario_FK (horario_id)		
horario_colaborador_n_doc_FK (nao_docente_id)		

Figura 56 - Tabela "horario\_colaborador".

### Tabela "uc\_curso":

Por último, em relação a estas tabelas de associação entre entidades principais, existe a tabela "uc\_curso", que na mais é do que o resultado da relação da tabela "curso" e "unidade\_curricular".

uc_curso		
PF*	curso_id	INTEGER
PF*	unidade_curricular_id	INTEGER
*	obrigatoria	CHAR (1)
*	semestre	INTEGER
*	ano	INTEGER
*	ects	INTEGER
*	presenca_obrigatoria	CHAR (1)
*	percentagem_presenca	NUMBER
*	created_at	TIMESTAMP
*	updated_at	TIMESTAMP
*	status	CHAR (1)
unidade_curricular_curso_PK (curso_id, unidade_curricular_id)		
uc_curso_curso_FK (curso_id)		

Figura 57 - Tabela "uc\_curso".

#### Entidades de Meta dados/Áreas:

Contando com apenas uma tabela, surgiu a necessidade haver uma tabela que guardasse dados das áreas científicas presentes na universidade.

Foi então que nasceu a tabela “área”.

#### Tabela “área”:

Esta tabela guarda dados relativos à área científica (por exemplo, matemática, informática, etc...).

area		
P	* id	INTEGER
	* nome	VARCHAR2 (50 CHAR)
	descricao	VARCHAR2 (255)
	* created_at	TIMESTAMP
	* updated_at	TIMESTAMP
	* status	CHAR (1)
area_PK (id)		

Figura 58 - Tabela "area".

#### Relacionamentos do Modelo Físico:

Ainda relativamente ao modelo físico, sabemos que é um modelo gigante e complexo, outrora não fosse a tentativa de representação real de um sistema de gestão de uma universidade.

Como tal, abaixo, segue uma tabela descritiva de todas as relações presentes no modelo, bem como as entidades que se relacionam e a cardinalidade das mesmas.

Entidade	Cardinalidade	Entidade Relacionada
andar	N:1	edificio
edificio	N:1	polo
espaco	N:1	andar
espaco	N:1	tipo_espaco
aula	N:1	espaco
aula	N:1	tipo_aula
docente_aula	N:1	docente
docente_aula	N:1	aula

Instituto Politécnico de Coimbra  
Instituto Superior de Engenharia de Coimbra

avaliacao	N:1	turma
avaliacao	N:1	tipo_avaliacao
avaliacao	N:1 (não obrigatório)	avaliacao (auto-relacionamento)
entrega	N:1	avaliacao
estudante_entrega	N:1	entrega
estudante_entrega	N:1	estudante
ficheiro_entrega	N:1	entrega
recurso	N:1	turma
recurso	N:1	docente
ficheiro_recurso	N:1	recurso
docente	N:1	departamento
nao_docente	N:1	departamento
nao_docente	N:1	cargo
gabinete_docente	N:1	docente
gabinete_docente	N:1	espaco
horario	N:1	turno
horario_colaborador	N:1	horario
horario_colaborador	N:1	nao_docente
inscricao	N:1	turma
inscricao	N:1	matricula
inscricao	N:1	estado_inscricao
matricula	N:1	curso
matricula	N:1	estado_matricula
matricula	N:1	estudante
propina	N:1	matricula
parcela_propina	N:1	propina
turma	N:1	unidade_curricular
turma	N:1	turno_aulas
turma_docente	N:1	turma
turma_docente	N:1	docente
turma_docente	N:1	funcao_docente
curso_area	N:1	curso
curso_area	N:1	area
uc_curso	N:1	curso
uc_curso	N:1	unidade_curricular
contacto_estudante	N:1	estudante
contacto_estudante	N:1	tipo_contacto
contacto_colaborador	N:1	colaborador
contacto_colaborador	N:1	tipo_contacto
nota	N:1	avaliacao
nota	N:1	Estudante
log	N:1	acao_log
log	N:1 (não obrigatório)	entidade_log

log	N:1	utilizador
-----	-----	------------

## Data Definition Language – DDL

Ora o DDL é um subconjunto da linguagem SQL, é usado para definir e gerir a estrutura dos objetos da base de dados, objetos esses que podem ser, tabelas, índices, vistas ou até mesmo esquemas.

Existem vários comandos DDL, mas os principais usados neste projeto são, o *CREATE* que tal como indica o nome, cria tabelas, o *ALTER* que também foi usado neste projeto e tal como nome diz, altera/modifica a estrutura de objetos já existentes, o *DROP* que remove objetos da base de dados e que também foi usado.

## Estrutura do DDL

Diz-se por aí que dar *DROP* nas tabelas antes de fazer qualquer tipo de operação é uma boa prática comum, isto porque evita duplicados, caso certo objeto já exista, além disso garante um controlo melhor sobre a estrutura, isto porque o utilizador sabe exatamente a estrutura que foi criada e não depende do estado anterior da base de dados. É também útil em ambientes de desenvolvimento/teste, onde o *script* é executado várias vezes e se pode dar *reset* ao estado.

E, como não podia deixar de ser, começou-se por fazer *DROP* a todas as tabelas.

```
8      DROP TABLE acao_log CASCADE CONSTRAINTS
9      ;
10
11     DROP TABLE andar CASCADE CONSTRAINTS
12     ;
13
14     DROP TABLE area CASCADE CONSTRAINTS
15     ;
16
17     DROP TABLE aula CASCADE CONSTRAINTS
18     ;
19
20     DROP TABLE avaliacao CASCADE CONSTRAINTS
21     ;
22
23     DROP TABLE cargo CASCADE CONSTRAINTS
24     ;
25
26     DROP TABLE colaborador CASCADE CONSTRAINTS
27     ;
```

Figura 59 - Drop nas Tabelas.

De referir ainda, que devido à extensão do *script* DDL (cerca de 1660 linhas), a figura 59, representa uma pequena parte das tabelas. O restante *script* acompanha o relatório aquando da sua submissão.

Logo após o *DROP*, foi possível arrancar para a criação da primeira tabela, que neste caso foi a tabela “acao\_log”, é importante relembrar que o DDL quando está a ser desenvolvido, normalmente começa-se a criar tabelas pela ordem de eventos e dependências. No caso, não foi isso que aconteceu, isto porque preferimos uma abordagem diferente (ordem alfabética) para não nos perdemos.

```
CREATE TABLE acao_log
(
    id          INTEGER NOT NULL ,
    nome        VARCHAR2 (20 CHAR) NOT NULL ,
    descricao   VARCHAR2 (20 CHAR) NOT NULL ,
    created_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status      CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE acao_log
ADD CONSTRAINT acao_log_PK PRIMARY KEY ( id ) ;
```

Figura 60 - Create e Alter Tabela "acao\_log".

Por fim, foi feito o *ALTER*, que por meio de uma restrição (*CONSTRAINT*), define a chave primária da tabela como sendo o campo “id”. Isto garante que esse campo contenha valores únicos e não nulos, assegurando a identificação exclusiva de cada registo.

De modo geral todas as tabelas presentes no nosso *script* DDL, foram criadas da mesma forma e com os seus respetivos atributos, alteradas/modificadas usando os mesmos comandos/cláusulas, pelo que se entende que mais explicações não serão propriamente necessárias.

```
CREATE TABLE andar
(
    id          INTEGER NOT NULL ,
    edificio_id INTEGER NOT NULL ,
    numero      INTEGER NOT NULL ,
    descricao   VARCHAR2 (50 CHAR) ,
    created_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status      CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE andar
ADD CONSTRAINT andar_PK PRIMARY KEY ( id ) ;
```

Figura 61 - Create e Alter da Tabela "andar".

```
CREATE TABLE area
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (50 CHAR) NOT NULL ,
    descricao     VARCHAR2 (255) ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE area
ADD CONSTRAINT area_PK PRIMARY KEY ( id ) ;
```

*Figura 62 - Create e Alter da Tabela "area".*

```
CREATE TABLE aula
(
    id            INTEGER NOT NULL ,
    data          DATE NOT NULL ,
    hora_inicio   DATE NOT NULL ,
    hora_fim      DATE NOT NULL ,
    sumario       VARCHAR2 (255 CHAR) ,
    espaco_id     INTEGER NOT NULL ,
    tipo_aula_id  INTEGER NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE aula
ADD CONSTRAINT aula_PK PRIMARY KEY ( id ) ;
```

*Figura 63 - Create e Alter da Tabela "aula".*



```
CREATE TABLE avaliacao
(
  id            INTEGER NOT NULL ,
  titulo        VARCHAR2 (100 CHAR) NOT NULL ,
  data          DATE NOT NULL ,
  data_entrega  DATE NOT NULL ,
  peso          NUMBER ,
  max_alunos    INTEGER NOT NULL ,
  turma_id      INTEGER NOT NULL ,
  tipo_avaliacao_id INTEGER NOT NULL ,
  avaliacao_pai_id INTEGER ,
  created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE avaliacao
ADD CONSTRAINT avaliacao_PK PRIMARY KEY ( id ) ;
```

Figura 64 - Create e Alter da Tabela "avaliacao".

```
CREATE TABLE cargo
(
  id            INTEGER NOT NULL ,
  nome          VARCHAR2 (50 CHAR) NOT NULL ,
  descricao     VARCHAR2 (255 CHAR) ,
  created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE cargo
ADD CONSTRAINT cargo_PK PRIMARY KEY ( id ) ;
```

Figura 65 - Create e Alter da Tabela "cargo".

```
CREATE TABLE colaborador
(
  id            INTEGER NOT NULL ,
  nome          VARCHAR2 (100 CHAR) NOT NULL ,
  data_contratacao DATE NOT NULL ,
  cc            VARCHAR2 (8 CHAR) ,
  nif           VARCHAR2 (9 CHAR) ,
  iban          VARCHAR2 (25 CHAR) ,
  created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE colaborador
ADD CONSTRAINT colaborador_PK PRIMARY KEY ( id ) ;
```

*Figura 66 - Create e Alter da Tabela "colaborador".*

```
CREATE TABLE contacto_colaborador
(
  contacto      VARCHAR2 (50 CHAR) NOT NULL ,
  tipo_contacto_id INTEGER NOT NULL ,
  colaborador_id INTEGER NOT NULL ,
  created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status        CHAR (1) DEFAULT '1' NOT NULL
)
;
```

*Figura 67 - Create da Tabela "contacto\_colaborador".*

```
CREATE TABLE contacto_estudante
(
    id            INTEGER NOT NULL ,
    contacto      VARCHAR2 (50 CHAR) NOT NULL ,
    tipo_contacto_id INTEGER NOT NULL ,
    estudante_id  INTEGER NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE contacto_estudante
ADD CONSTRAINT contacto_PK PRIMARY KEY ( id ) ;
```

Figura 68 - Create e Alter da Tabela "contacto\_estudante".

```
CREATE TABLE curso
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (100 CHAR) NOT NULL ,
    codigo        VARCHAR2 (20 CHAR) NOT NULL ,
    descricao     VARCHAR2 (255 CHAR) NOT NULL ,
    duracao       INTEGER NOT NULL ,
    ects          INTEGER NOT NULL ,
    tipo_curso_id INTEGER NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE curso
ADD CONSTRAINT curso_PK PRIMARY KEY ( id ) ;
```

Figura 69 - Create e Alter da Tabela "curso".

```
CREATE TABLE curso_area
(
    area_id    INTEGER NOT NULL ,
    curso_id   INTEGER NOT NULL ,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status     CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE curso_area
ADD CONSTRAINT curso_area_PK PRIMARY KEY ( area_id, curso_id ) ;
```

*Figura 70 - Create e Alter da Tabela "curso\_area".*

```
CREATE TABLE departamento
(
    id                INTEGER NOT NULL ,
    nome              VARCHAR2 (30 CHAR) NOT NULL ,
    descricao         VARCHAR2 (100 CHAR) ,
    email             VARCHAR2 (50 CHAR) ,
    telefone          VARCHAR2 (20 CHAR) ,
    tipo_departamento_id INTEGER NOT NULL ,
    created_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE departamento
ADD CONSTRAINT departamento_PK PRIMARY KEY ( id ) ;
```

*Figura 71 - Create e Alter da Tabela "departamento".*

```
CREATE TABLE docente
(
    id                INTEGER NOT NULL ,
    departamento_id   INTEGER NOT NULL ,
    created_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE docente
ADD CONSTRAINT docente_PK PRIMARY KEY ( id ) ;
```

*Figura 72 - Create e Alter da Tabela "docente".*

```
CREATE TABLE docente_aula
(
    aula_id          INTEGER NOT NULL ,
    docente_id       INTEGER NOT NULL ,
    created_at       TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at       TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status           CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE docente_aula
ADD CONSTRAINT docente_aula_PK PRIMARY KEY ( aula_id, docente_id ) ;
```

*Figura 73 - Create e Alter da Tabela "docente\_aula".*

```
CREATE TABLE edificio
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (50 CHAR) NOT NULL ,
    polo_id       INTEGER NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE edificio
ADD CONSTRAINT edificio_PK PRIMARY KEY ( id ) ;
```

Figura 74 - Create e Alter da Tabela "edificio".

```
CREATE TABLE entidade_log
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (50 CHAR) NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE entidade_log
ADD CONSTRAINT entidade_log_PK PRIMARY KEY ( id ) ;
```

Figura 75 - Create e Alter da Tabela "entidade\_log".

```
CREATE TABLE entrega
(
    id            INTEGER NOT NULL ,
    data_entrega  DATE NOT NULL ,
    avaliacao_id  INTEGER NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE entrega
ADD CONSTRAINT entrega_PK PRIMARY KEY ( id ) ;
```

Figura 76 - Create e Alter da Tabela "entrega".

```
CREATE TABLE espaco
(
  id            INTEGER NOT NULL ,
  andar_id     INTEGER NOT NULL ,
  tipo_espaco_id INTEGER NOT NULL ,
  nome         VARCHAR2 (50 CHAR) NOT NULL ,
  capacidade   INTEGER NOT NULL ,
  created_at   TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at   TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status       CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE espaco
ADD CONSTRAINT espaco_PK PRIMARY KEY ( id ) ;
```

Figura 77 - Create e Alter da Tabela "espaco".

```
CREATE TABLE estado_inscricao
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (50 CHAR) NOT NULL ,
    descricao     VARCHAR2 (255 CHAR) NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE estado_inscricao
ADD CONSTRAINT estado_inscricao_PK PRIMARY KEY ( id ) ;
```

Figura 78 - Create e Alter da Tabela "estado\_inscricao".

```
CREATE TABLE estado_matricula
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (50 CHAR) NOT NULL ,
    descricao     VARCHAR2 (255 CHAR) ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE estado_matricula
ADD CONSTRAINT estado_matricula_PK PRIMARY KEY ( id ) ;
```

Figura 79 - Create e Alter da Tabela "estado\_matricula".



```
CREATE TABLE estudante
(
  id            INTEGER NOT NULL ,
  nome          VARCHAR2 (50 CHAR) NOT NULL ,
  morada        VARCHAR2 (100 CHAR) ,
  data_nascimento DATE NOT NULL ,
  cc            VARCHAR2 (8) ,
  nif           VARCHAR2 (9) ,
  iban          VARCHAR2 (25 CHAR) ,
  created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE estudante
ADD CONSTRAINT estudante_PK PRIMARY KEY ( id ) ;
```

Figura 80 - Create e Alter da Tabela "estudante".

```
CREATE TABLE estudante_entrega
(
  estudante_id INTEGER NOT NULL ,
  entrega_id   INTEGER NOT NULL ,
  created_at   TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at   TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status       CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE estudante_entrega
ADD CONSTRAINT estudante_entrega_PK PRIMARY KEY ( estudante_id, entrega_id ) ;
```

Figura 81 - Create e Alter da Tabela "estudante\_entrega".

```
CREATE TABLE ficheiro_entrega
(
  id          INTEGER NOT NULL ,
  entrega_id  INTEGER NOT NULL ,
  nome        VARCHAR2 (100 CHAR) NOT NULL ,
  ficheiro    BLOB NOT NULL ,
  tamanho     INTEGER NOT NULL ,
  tipo        VARCHAR2 (20 CHAR) NOT NULL ,
  created_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status      CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE ficheiro_entrega
ADD CONSTRAINT ficheiro_entrega_PK PRIMARY KEY ( id ) ;
```

*Figura 82 - Create e Alter da Tabela "ficheiro\_entrega".*

```
CREATE TABLE ficheiro_recurso
(
  id          INTEGER NOT NULL ,
  recurso_id  INTEGER NOT NULL ,
  nome        VARCHAR2 (100 CHAR) NOT NULL ,
  ficheiro    BLOB NOT NULL ,
  created_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status      CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE ficheiro_recurso
ADD CONSTRAINT ficheiro_recurso_PK PRIMARY KEY ( id ) ;
```

*Figura 83 - Create e Alter da Tabela "ficheiro\_recurso".*

```
CREATE TABLE funcao_docente
(
  id          INTEGER NOT NULL ,
  nome        VARCHAR2 (50 CHAR) NOT NULL ,
  descricao   VARCHAR2 (255 CHAR) ,
  created_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status      CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE funcao_docente
ADD CONSTRAINT funcao_docente_PK PRIMARY KEY ( id ) ;
```

*Figura 84 - Create e Alter da Tabela "funcao\_docente".*

```
CREATE TABLE gabinete_docente
(
    espaco_id INTEGER NOT NULL ,
    docente_id INTEGER NOT NULL ,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE gabinete_docente
ADD CONSTRAINT gabinete_docente_PK PRIMARY KEY ( espaco_id, docente_id ) ;
```

Figura 85 - Create e Alter da Tabela "gabinete\_docente".

```
CREATE TABLE horario
(
    id INTEGER NOT NULL ,
    data DATE NOT NULL ,
    turno_id INTEGER NOT NULL ,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE horario
ADD CONSTRAINT horario_PK PRIMARY KEY ( id ) ;
```

Figura 86 - Create e Alter da Tabela "horario".

```
CREATE TABLE horario_colaborador
(
    horario_id INTEGER NOT NULL ,
    nao_docente_id INTEGER NOT NULL ,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE horario_colaborador
ADD CONSTRAINT horario_colaborador_PK PRIMARY KEY ( horario_id, nao_docente_id ) ;
```

Figura 87 - Create e Alter da Tabela "horario\_colaborador".

```
CREATE TABLE inscricao
(
    turma_id          INTEGER NOT NULL ,
    matricula_id      INTEGER NOT NULL ,
    data              DATE NOT NULL ,
    estado_inscricao_id INTEGER NOT NULL ,
    created_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE inscricao
ADD CONSTRAINT inscricao_PK PRIMARY KEY ( matricula_id, turma_id ) ;
```

*Figura 88 - Create e Alter da Tabela "inscricao".*

```
CREATE TABLE log
(
    id                INTEGER NOT NULL ,
    data              CLOB ,
    entidade_log_id   INTEGER ,
    acao_log_id       INTEGER NOT NULL ,
    utilizador_id     INTEGER NOT NULL ,
    created_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE log
ADD CONSTRAINT log_PK PRIMARY KEY ( id ) ;
```

*Figura 89 - Create Alter da Tabela "log".*

```
CREATE TABLE matricula
(
    id                INTEGER NOT NULL ,
    curso_id          INTEGER NOT NULL ,
    ano_inscricao     DATE ,
    estado_matricula_id INTEGER NOT NULL ,
    estudante_id       INTEGER NOT NULL ,
    created_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE matricula
ADD CONSTRAINT matricula_PK PRIMARY KEY ( id ) ;
```

*Figura 90 - Create e Alter da Tabela "matricula".*

```
CREATE TABLE nao_docente
(
    id                INTEGER NOT NULL ,
    departamento_id   INTEGER NOT NULL ,
    cargo_id          INTEGER NOT NULL ,
    created_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE nao_docente
ADD CONSTRAINT nao_docente_PK PRIMARY KEY ( id ) ;
```

*Figura 91 - Create e Alter da Tabela "nao\_docente".*

```
CREATE TABLE nota
(
    estudante_id INTEGER NOT NULL ,
    avaliacao_id INTEGER NOT NULL ,
    nota         NUMBER NOT NULL ,
    comentario   VARCHAR2 (255 CHAR) ,
    created_at   TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at   TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status       CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE nota
ADD CONSTRAINT nota_PK PRIMARY KEY ( avaliacao_id, estudante_id ) ;
```

*Figura 92 - Create e Alter da Tabela "nota".*

```
CREATE TABLE parcela_propina
(
    id            INTEGER NOT NULL ,
    valor         NUMBER NOT NULL ,
    data_vencimento DATE NOT NULL ,
    data_pagamento DATE ,
    numero        INTEGER NOT NULL ,
    propina_id    INTEGER NOT NULL ,
    estado        CHAR (1) NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE parcela_propina
ADD CONSTRAINT parcela_propina_PK PRIMARY KEY ( id ) ;
```

*Figura 93 - Create e Alter da Tabela "parcela\_propina".*

```
CREATE TABLE polo
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (50 CHAR) NOT NULL ,
    morada        VARCHAR2 (100 CHAR) NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE polo
ADD CONSTRAINT polo_PK PRIMARY KEY ( id ) ;
```

*Figura 94 - Create e Alter da Tabela "polo".*

```
CREATE TABLE presenca
(
    aula_id       INTEGER NOT NULL ,
    estudante_id  INTEGER NOT NULL ,
    presente      CHAR (1) NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE presenca
ADD CONSTRAINT presenca_PK PRIMARY KEY ( aula_id, estudante_id ) ;
```

*Figura 95 - Create e Alter da Tabela "presenca".*



```
CREATE TABLE propina
(
    id                INTEGER NOT NULL ,
    matricula_id      INTEGER NOT NULL ,
    numero_parcelas    INTEGER NOT NULL ,
    estado            CHAR (1) NOT NULL ,
    created_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE propina
ADD CONSTRAINT propina_PK PRIMARY KEY ( id ) ;
```

Figura 96 - Create e Alter da Tabela "propina".

```
CREATE TABLE recurso
(
    id                INTEGER NOT NULL ,
    nome              VARCHAR2 (100 CHAR) NOT NULL ,
    descricao         VARCHAR2 (255 CHAR) ,
    turma_id          INTEGER NOT NULL ,
    docente_id        INTEGER NOT NULL ,
    created_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE recurso
ADD CONSTRAINT recurso_PK PRIMARY KEY ( id ) ;
```

Figura 97 - Create e Alter da Tabela "recurso".

```
CREATE TABLE tipo_aula
(
    id                INTEGER NOT NULL ,
    nome              VARCHAR2 (50 CHAR) NOT NULL ,
    created_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE tipo_aula
ADD CONSTRAINT tipo_aula_PK PRIMARY KEY ( id ) ;
```

Figura 98 - Create e Alter da Tabela "tipo\_aula".

Instituto Politécnico de Coimbra  
Instituto Superior de Engenharia de Coimbra

```
CREATE TABLE tipo_avaliacao
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (50 CHAR) NOT NULL ,
    descricao     VARCHAR2 (255 CHAR) ,
    requer_entrega CHAR (1) NOT NULL ,
    permite_grupo CHAR (1) NOT NULL ,
    permite_filhos CHAR (1) NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE tipo_avaliacao
ADD CONSTRAINT tipo_avaliacao_PK PRIMARY KEY ( id ) ;
```

Figura 99 - Create e Alter da Tabela "tipo\_avaliacao".

```
CREATE TABLE tipo_contacto
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (50 CHAR) NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE tipo_contacto
ADD CONSTRAINT tipo_contacto_PK PRIMARY KEY ( id ) ;
```

Figura 100 - Create e Alter da Tabela "tipo\_contacto".

```
CREATE TABLE tipo_curso
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (50 CHAR) NOT NULL ,
    valor_propinas NUMBER NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE tipo_curso
ADD CONSTRAINT tipo_curso_PK PRIMARY KEY ( id ) ;
```

Figura 101 - Create e Alter da Tabela "tipo\_curso".

```
CREATE TABLE tipo_departamento
(
  id          INTEGER NOT NULL ,
  nome        VARCHAR2 (30 CHAR) NOT NULL ,
  descricao   VARCHAR2 (255 CHAR) ,
  created_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status      CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE tipo_departamento
ADD CONSTRAINT tipo_departamento_PK PRIMARY KEY ( id ) ;
```

*Figura 102 - Create e Alter da Tabela "tipo\_departamento".*

```
CREATE TABLE tipo_espaco
(
  id          INTEGER NOT NULL ,
  nome        VARCHAR2 (50 CHAR) NOT NULL ,
  permite_aula CHAR (1) NOT NULL ,
  created_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at  TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status      CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE tipo_espaco
ADD CONSTRAINT tipo_espaco_PK PRIMARY KEY ( id ) ;
```

*Figura 103 - Create e Alter da Tabela "tipo\_espaco".*

```
CREATE TABLE turma
(
  id                INTEGER NOT NULL ,
  nome              VARCHAR2 (20 CHAR) NOT NULL ,
  ano_letivo        VARCHAR2 (10 CHAR) NOT NULL ,
  unidade_curricular_id INTEGER NOT NULL ,
  turno_aulas_id   INTEGER NOT NULL ,
  created_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE turma
ADD CONSTRAINT turma_PK PRIMARY KEY ( id ) ;
```

*Figura 104 - Create e Alter da Tabela "turma".*

```
CREATE TABLE turma_docente
(
  turma_id          INTEGER NOT NULL ,
  funcao_docente_id INTEGER NOT NULL ,
  docente_id        INTEGER NOT NULL ,
  created_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  updated_at        TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
  status            CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE turma_docente
ADD CONSTRAINT turma_docente_PK PRIMARY KEY ( turma_id ) ;
```

*Figura 105 - Create e Alter da Tabela "turma\_docente".*

```
CREATE TABLE turno
(
    id            INTEGER NOT NULL ,
    hora_inicio  DATE  NOT NULL ,
    hora_fim     DATE  NOT NULL ,
    created_at   TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at   TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status       CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE turno
ADD CONSTRAINT turno_PK PRIMARY KEY ( id ) ;
```

*Figura 106 - Create e Alter da Tabela "turno".*

```
CREATE TABLE turno_aulas
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (50 CHAR) NOT NULL ,
    hora_entrada  INTEGER ,
    hora_saida    INTEGER ,
    created_at   TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at   TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status       CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE turno_aulas
ADD CONSTRAINT turno_aulas_PK PRIMARY KEY ( id ) ;
```

*Figura 107 - Create e Alter da Tabela "turno\_aulas".*

```
CREATE TABLE uc_curso
(
    curso_id            INTEGER NOT NULL ,
    unidade_curricular_id INTEGER NOT NULL ,
    obrigatoria         CHAR (1) NOT NULL ,
    semestre            INTEGER NOT NULL ,
    ano                 INTEGER NOT NULL ,
    ects                INTEGER NOT NULL ,
    presenca_obrigatoria CHAR (1) NOT NULL ,
    percentagem_presenca NUMBER ,
    created_at          TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at          TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status              CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE uc_curso
ADD CONSTRAINT unidade_curricular_curso_PK PRIMARY KEY ( curso_id, unidade_curricular_id ) ;
```

Figura 108 - Create e Alter da Tabela "uc\_curso".

```
CREATE TABLE uc_departamento
(
    departamento_id     INTEGER NOT NULL ,
    unidade_curricular_id INTEGER NOT NULL ,
    created_at          TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at          TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status              CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE uc_departamento
ADD CONSTRAINT uc_departamento_PK PRIMARY KEY ( departamento_id, unidade_curricular_id ) ;
```

Figura 109 - Create e Alter da Tabela "uc\_departamento".

```
CREATE TABLE uc_docente
(
    unidade_curricular_id INTEGER NOT NULL ,
    docente_id            INTEGER NOT NULL ,
    funcao                VARCHAR2 (30 CHAR) ,
    created_at            TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at            TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status                CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE uc_docente
ADD CONSTRAINT unidade_curricular_docente_PK PRIMARY KEY ( unidade_curricular_id, docente_id ) ;
```

Figura 110 - Create e Alter da Tabela "uc\_docente".

```
CREATE TABLE unidade_curricular
(
    id            INTEGER NOT NULL ,
    nome          VARCHAR2 (100 CHAR) NOT NULL ,
    codigo        VARCHAR2 (20 CHAR) NOT NULL ,
    horas_teoricas INTEGER NOT NULL ,
    horas_praticas INTEGER NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE unidade_curricular
ADD CONSTRAINT unidade_curricular_PK PRIMARY KEY ( id ) ;
```

*Figura 111 - Create e Alter da Tabela "unidade\_curricular".*

```
CREATE TABLE utilizador
(
    id            INTEGER NOT NULL ,
    email         VARCHAR2 (50 CHAR) NOT NULL ,
    password      VARCHAR2 (50 CHAR) NOT NULL ,
    created_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    updated_at    TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL ,
    status        CHAR (1) DEFAULT '1' NOT NULL
)
;

ALTER TABLE utilizador
ADD CONSTRAINT utilizador_PK PRIMARY KEY ( id ) ;
```

*Figura 112 - Create e Alter da Tabela "utilizador".*

Como referido no início da explicação deste tópico, foi dito que de tabela em tabela o mecanismo de criação e alteração de tabelas, era exatamente o mesmo, definitivamente após estas figuras todas, de facto é possível comprovar isso mesmo.

Após a criação das tabelas e dos demais processos atrelados à criação das mesmas, agora é necessário continuar a fazer *ALTER TABLE* mas desta vez por causa das chaves estrangeiras.

```
ALTER TABLE andar
  ADD CONSTRAINT andar_edificio_FK FOREIGN KEY
  (
    edificio_id
  )
  REFERENCES edificio
  (
    id
  )
;
```

Figura 113 - Alter da Tabela "andar".

```
ALTER TABLE aula
  ADD CONSTRAINT aula_espaco_FK FOREIGN KEY
  (
    espaco_id
  )
  REFERENCES espaco
  (
    id
  )
;
```

```
ALTER TABLE aula
  ADD CONSTRAINT aula_tipo_aula_FK FOREIGN KEY
  (
    tipo_aula_id
  )
  REFERENCES tipo_aula
  (
    id
  )
;
```

Figura 114 - Alter da Tabela "aula".



```
ALTER TABLE avaliacao
ADD CONSTRAINT avaliacao_avaliacao_FK FOREIGN KEY
(
    avaliacao_pai_id
)
REFERENCES avaliacao
(
    id
)
;

ALTER TABLE avaliacao
ADD CONSTRAINT avaliacao_tipo_avaliacao_FK FOREIGN KEY
(
    tipo_avaliacao_id
)
REFERENCES tipo_avaliacao
(
    id
)
;

ALTER TABLE avaliacao
ADD CONSTRAINT avaliacao_turma_FK FOREIGN KEY
(
    turma_id
)
REFERENCES turma
(
    id
)
;
```

*Figura 115 - Alter da Tabela "avaliacao".*

```
ALTER TABLE colaborador
ADD CONSTRAINT colaborador_utilizador_FK FOREIGN KEY
(
    id
)
REFERENCES utilizador
(
    id
)
;
```

*Figura 116 - Alter da Tabela "colaborador".*

```
ALTER TABLE contacto_estudante
ADD CONSTRAINT cont_estudante_tipo_FK FOREIGN KEY
(
    tipo_contacto_id
)
REFERENCES tipo_contacto
(
    id
)
;
```

Figura 117 - Alter da Tabela "contacto\_estudante".

```
ALTER TABLE curso_area
ADD CONSTRAINT curso_area_area_FK FOREIGN KEY
(
    area_id
)
REFERENCES area
(
    id
)
;
```

```
ALTER TABLE curso_area
ADD CONSTRAINT curso_area_curso_FK FOREIGN KEY
(
    curso_id
)
REFERENCES curso
(
    id
)
;
```

Figura 118 - Alter da Tabela "curso\_area".

```
ALTER TABLE curso
ADD CONSTRAINT curso_tipo_curso_FK FOREIGN KEY
(
    tipo_curso_id
)
REFERENCES tipo_curso
(
    id
)
;
```

Figura 119 - Alter da Tabela "curso".

```
ALTER TABLE departamento
ADD CONSTRAINT depart_tipo_depart_FK FOREIGN KEY
(
    tipo_departamento_id
)
REFERENCES tipo_departamento
(
    id
)
;
```

Figura 120 - Alter da Tabela "departamento".

```
ALTER TABLE docente_aula
ADD CONSTRAINT docente_aula_aula_FK FOREIGN KEY
(
    aula_id
)
REFERENCES aula
(
    id
)
;
```

```
ALTER TABLE docente_aula
ADD CONSTRAINT docente_aula_docente_FK FOREIGN KEY
(
    docente_id
)
REFERENCES docente
(
    id
)
;
```

Figura 121 - Alter da Tabela "docente\_aula".

```
ALTER TABLE docente
  ADD CONSTRAINT docente_colaborador_FK FOREIGN KEY
  (
    id
  )
  REFERENCES colaborador
  (
    id
  )
;

ALTER TABLE docente
  ADD CONSTRAINT docente_departamento_FK FOREIGN KEY
  (
    departamento_id
  )
  REFERENCES departamento
  (
    id
  )
;
```

*Figura 122 - Alter da Tabela "docente".*

```
ALTER TABLE edificio
  ADD CONSTRAINT edificio_polo_FK FOREIGN KEY
  (
    polo_id
  )
  REFERENCES polo
  (
    id
  )
;
```

*Figura 123 - Alter da Tabela "edificio".*

```
ALTER TABLE entrega
ADD CONSTRAINT entrega_avaliacao_FK FOREIGN KEY
(
    avaliacao_id
)
REFERENCES avaliacao
(
    id
)
;
```

*Figura 124 - Alter da Tabela "entrega".*

```
ALTER TABLE espaco
ADD CONSTRAINT espaco_tipo_espaco_FK FOREIGN KEY
(
    tipo_espaco_id
)
REFERENCES tipo_espaco
(
    id
)
;
```

*Figura 125 - Alter da Tabela "espaco".*

Instituto Politécnico de Coimbra  
Instituto Superior de Engenharia de Coimbra

```
ALTER TABLE estudante_entrega
ADD CONSTRAINT estudante_entrega_entrega_FK FOREIGN KEY
(
    entrega_id
)
REFERENCES entrega
(
    id
)
;

ALTER TABLE estudante_entrega
ADD CONSTRAINT estudante_entrega_estudante_FK FOREIGN KEY
(
    estudante_id
)
REFERENCES estudante
(
    id
)
;
```

*Figura 126 - Alter da Tabela "estudante\_entrega".*

```
ALTER TABLE estudante
ADD CONSTRAINT estudante_utilizador_FK FOREIGN KEY
(
    id
)
REFERENCES utilizador
(
    id
)
;
```

*Figura 127 - Alter da Tabela "estudante".*

```
ALTER TABLE ficheiro_entrega
  ADD CONSTRAINT ficheiro_entrega_FK FOREIGN KEY
  (
    entrega_id
  )
  REFERENCES entrega
  (
    id
  )
;

ALTER TABLE ficheiro_recurso
  ADD CONSTRAINT ficheiro_recurso_recurso_FK FOREIGN KEY
  (
    recurso_id
  )
  REFERENCES recurso
  (
    id
  )
;
```

Figura 128 - Alter das Tabelas "ficheiro\_entrega" e "ficheiro\_recurso".

```
ALTER TABLE gabinete_docente
  ADD CONSTRAINT gabinete_docente_docente_FK FOREIGN KEY
  (
    docente_id
  )
  REFERENCES docente
  (
    id
  )
;

ALTER TABLE gabinete_docente
  ADD CONSTRAINT gabinete_docente_espaco_FK FOREIGN KEY
  (
    espaco_id
  )
  REFERENCES espaco
  (
    id
  )
;
```

Figura 129 - Alter da Tabela "gabinete\_docente".

```
ALTER TABLE horario_colaborador
  ADD CONSTRAINT horario_colaborador_horario_FK FOREIGN KEY
  (
    horario_id
  )
  REFERENCES horario
  (
    id
  )
;

ALTER TABLE horario_colaborador
  ADD CONSTRAINT horario_colaborador_n_doc_FK FOREIGN KEY
  (
    nao_docente_id
  )
  REFERENCES nao_docente
  (
    id
  )
;
```

*Figura 130 - Alter da Tabela "horario\_colaborador".*

```
ALTER TABLE horario
  ADD CONSTRAINT horario_turno_FK FOREIGN KEY
  (
    turno_id
  )
  REFERENCES turno
  (
    id
  )
;
```

*Figura 131 - Alter da Tabela "horario".*



```
ALTER TABLE matricula
  ADD CONSTRAINT inscricao_curso_curso_FK FOREIGN KEY
  (
    curso_id
  )
  REFERENCES curso
  (
    id
  )
;

ALTER TABLE matricula
  ADD CONSTRAINT inscricao_curso_estudante_FK FOREIGN KEY
  (
    estudante_id
  )
  REFERENCES estudante
  (
    id
  )
;
```

*Figura 132 - Alter da Tabela "matricula".*

Instituto Politécnico de Coimbra  
Instituto Superior de Engenharia de Coimbra

```
ALTER TABLE inscricao
ADD CONSTRAINT inscricao_estado_inscricao_FK FOREIGN KEY
(
    estado_inscricao_id
)
REFERENCES estado_inscricao
(
    id
)
;

ALTER TABLE inscricao
ADD CONSTRAINT inscricao_turma_matricula_FK FOREIGN KEY
(
    matricula_id
)
REFERENCES matricula
(
    id
)
;

ALTER TABLE inscricao
ADD CONSTRAINT inscricao_turma_turma_FK FOREIGN KEY
(
    turma_id
)
REFERENCES turma
(
    id
)
;
```

*Figura 133 - Alter da Tabela "inscricao".*

```
ALTER TABLE log
ADD CONSTRAINT log_acao_log_FK FOREIGN KEY
(
    acao_log_id
)
REFERENCES acao_log
(
    id
)
;

ALTER TABLE log
ADD CONSTRAINT log_entidade_log_FK FOREIGN KEY
(
    entidade_log_id
)
REFERENCES entidade_log
(
    id
)
;

ALTER TABLE log
ADD CONSTRAINT log_utilizador_FK FOREIGN KEY
(
    utilizador_id
)
REFERENCES utilizador
(
    id
)
;
```

Figura 134 - Alter da Tabela "log".

```
ALTER TABLE matricula
ADD CONSTRAINT matricula_estado_matricula_FK FOREIGN KEY
(
    estado_matricula_id
)
REFERENCES estado_matricula
(
    id
)
;
```

Figura 135 - Outro Alter da Tabela "matricula".

```
ALTER TABLE nao_docente
ADD CONSTRAINT nao_docente_cargo_FK FOREIGN KEY
(
    cargo_id
)
REFERENCES cargo
(
    id
)
;

ALTER TABLE nao_docente
ADD CONSTRAINT nao_docente_colaborador_FK FOREIGN KEY
(
    id
)
REFERENCES colaborador
(
    id
)
;

ALTER TABLE nao_docente
ADD CONSTRAINT nao_docente_departamento_FK FOREIGN KEY
(
    departamento_id
)
REFERENCES departamento
(
    id
)
;
```

*Figura 136 - Alter da Tabela "nao\_docente".*

```
ALTER TABLE nota
ADD CONSTRAINT nota_avaliacao_FK FOREIGN KEY
(
    avaliacao_id
)
REFERENCES avaliacao
(
    id
)
;

ALTER TABLE nota
ADD CONSTRAINT nota_estudante_FK FOREIGN KEY
(
    estudante_id
)
REFERENCES estudante
(
    id
)
;
```

*Figura 137 - Alter da Tabela "nota".*

```
ALTER TABLE parcela_propina
ADD CONSTRAINT parcela_propina_propina_FK FOREIGN KEY
(
    propina_id
)
REFERENCES propina
(
    id
)
;
```

Figura 138 - Alter da Tabela "parcela\_propina".

```
ALTER TABLE presenca
ADD CONSTRAINT presenca_aula_FK FOREIGN KEY
(
    aula_id
)
REFERENCES aula
(
    id
)
;
```

```
ALTER TABLE presenca
ADD CONSTRAINT presenca_estudante_FK FOREIGN KEY
(
    estudante_id
)
REFERENCES estudante
(
    id
)
;
```

Figura 139 - Alter da Tabela "presenca".

```
ALTER TABLE propina
ADD CONSTRAINT propina_matricula_FK FOREIGN KEY
(
    matricula_id
)
REFERENCES matricula
(
    id
)
;
```

Figura 140 - Alter da Tabela Propina.

```
ALTER TABLE recurso
  ADD CONSTRAINT recurso_docente_FK FOREIGN KEY
  (
    docente_id
  )
  REFERENCES docente
  (
    id
  )
;

ALTER TABLE recurso
  ADD CONSTRAINT recurso_turma_FK FOREIGN KEY
  (
    turma_id
  )
  REFERENCES turma
  (
    id
  )
;
```

*Figura 141 - Alter da Tabela "recurso".*

```
ALTER TABLE turma_docente
  ADD CONSTRAINT turma_docente_docente_FK FOREIGN KEY
  (
    docente_id
  )
  REFERENCES docente
  (
    id
  )
;

ALTER TABLE turma_docente
  ADD CONSTRAINT turma_docente_funcao_FK FOREIGN KEY
  (
    funcao_docente_id
  )
  REFERENCES funcao_docente
  (
    id
  )
;
```

*Figura 142 - Alter da Tabela "turma\_docente".*

```
ALTER TABLE espaco
ADD CONSTRAINT sala_andar_FK FOREIGN KEY
(
    andar_id
)
REFERENCES andar
(
    id
)
;
```

*Figura 143 - Outro Alter da Tabela "espaco".*

```
ALTER TABLE turma
ADD CONSTRAINT turma_turno_aulas_FK FOREIGN KEY
(
    turno_aulas_id
)
REFERENCES turno_aulas
(
    id
)
;
```

```
ALTER TABLE turma
ADD CONSTRAINT turma_unidade_curricular_FK FOREIGN KEY
(
    unidade_curricular_id
)
REFERENCES unidade_curricular
(
    id
)
;
```

*Figura 144 - Alter da Tabela "turma".*

```
ALTER TABLE uc_curso
ADD CONSTRAINT uc_curso_curso_FK FOREIGN KEY
(
    curso_id
)
REFERENCES curso
(
    id
)
;

ALTER TABLE uc_curso
ADD CONSTRAINT uc_curso_uc_FK FOREIGN KEY
(
    unidade_curricular_id
)
REFERENCES unidade_curricular
(
    id
)
;
```

*Figura 145 - Alter da Tabela "uc\_curso".*



```
ALTER TABLE uc_departamento
ADD CONSTRAINT uc_depart_depart_FK FOREIGN KEY
(
    departamento_id
)
REFERENCES departamento
(
    id
)
;

ALTER TABLE uc_departamento
ADD CONSTRAINT uc_depart_uc_FK FOREIGN KEY
(
    unidade_curricular_id
)
REFERENCES unidade_curricular
(
    id
)
;
```

Figura 146 - Alter da Tabela "uc\_departamento".

```
ALTER TABLE uc_docente
  ADD CONSTRAINT uc_docente_docente_FK FOREIGN KEY
  (
    docente_id
  )
  REFERENCES docente
  (
    id
  )
;

ALTER TABLE turma_docente
  ADD CONSTRAINT uc_docente_turma_FK FOREIGN KEY
  (
    turma_id
  )
  REFERENCES turma
  (
    id
  )
;

ALTER TABLE uc_docente
  ADD CONSTRAINT uc_docente_uc_FK FOREIGN KEY
  (
    unidade_curricular_id
  )
  REFERENCES unidade_curricular
  (
    id
  )
;
```

Figura 147 - Alter da Tabela "uc\_docente".

Desta forma, se explica um bocado mais facilmente os *ALTER TABLE*, bem como as cedências de chaves estrangeiras por parte de certas entidades a outras entidades.

## Inserção de Dados

Devido à extensão do nosso sistema, nós optámos por fazer apenas duzentas inserções (*INSERT*) por entidade.

Como já referido anteriormente, a ferramenta usada para esta ocasião foi o *Mockaroo*.

Tivemos alguns problemas na inserção, tipicamente por que a ferramenta está em inglês e, portanto, gera dados falsos em inglês e sem estarem totalmente alienados com os atributos, como por exemplo, nos atributos que implicam introduzir nomes, a ferramenta *Mockaroo* achou por bem dar nomes de plantas e, portanto, este é um dos pequenos exemplos do quão mal a parte da transformação dos dados funcionou.

## Conclusão

A conclusão será mais a título pessoal do grupo, se bem que em certas situações ao longo do documento, fomos intervindo para justificar certas coisas relativas ao projeto, deixando a escrita formal de lado.

Não consideramos de todo um projeto difícil, isto porque já tínhamos passado pela mesma situação no ano passado na disciplina de Bases de Dados I. O que custou mais e foi mais trabalhoso, foi a modelação em *Data Modeler* de um projeto tão extenso.

Ainda assim, fizemos o trabalho sabendo da proporção do mesmo e estamos prontos para acarretar com as possíveis consequências disso na próxima e última fase deste projeto, quando começarmos a englobar *triggers*, funções, procedimentos e cursores.

Podemos dizer que independentemente da apreciação que este nosso projeto possa vir a receber, nos esforçamos, soubemos dividir bem as tarefas, sabemos o que está feito e o que pode ter corrido menos bem e isso é certamente motivo de orgulho.

P.S. Este documento e consequente projeto dispensa a declaração de *webgrafia* ou bibliografia, visto que o presente documento, apenas explica a execução passo a passo do projeto.

Ademais é importante revelar, que este documento, foi submetido juntamente com os restantes ficheiros que complementam o projeto (modelo físico, *script* ddl e posterior ficheiro com os diversos dados inseridos no sistema de ensino superior).