Universidade do Minho
Escola de Engenharia
Mestrado Integrado em Engenharia Informática

Unidade Curricular de Administração e Exploração de Bases de Dados

Ano Letivo de 2017/2018

Relatório do Trabalho Prático

Bruno Pereira (a75135), João Almeida (a75209),

Luís Fernandes (a74748), Maria Ana de Brito (a73580)

Janeiro, 2018

AEBD

Índice

Índice de Figuras

Índice de Tabelas

1. Introdução

Uma Base de Dados possui imensas informações a que temos acesso, de forma a poder monitorizar o seu estado e desempenho. Deste modo, com este projeto prático visamos a análise das várias vertentes com o objetivo de apresentar os principais parâmetros de avaliação da *performance* de uma BD.

Selecionaram-se, assim, os aspetos mais relevantes para esta análise disponibilizados pelas *views* de administração. De seguida, desenvolveu-se um programa que recolhesse esta informação e a integrasse numa Base de Dados *Oracle*. Esta foi concebida tendo em conta os dados que seriam extraídos das vistas anteriormente utilizadas – seguiram-se, pois, todos os passos desde a criação de um modelo concetual, contendo as entidades, seus atributos e relacionamentos, até à implementação física.

Por fim, tratou-se de criar uma API em *REST* com o objetivo de se conectar à Base de Dados *Oracle* previamente criada. Recolhendo os dados lá armazenados, devolve-os no formato *JSON*. Estes resultados são apresentados numa página *HTML*, ilustrando o desempenho da Base de Dados segundo uma dimensão temporal.

Área de Aplicação: Administração de Base de Dados, Base de Dados, Monitorização
Palavras-Chave: base de dados, dados, informação, desempenho, *CPU*, memória, utilizador, *session*, *role*, *privilege*, *tablespace*, *datafile*

2. Modelo Conceptual da Base de Dados

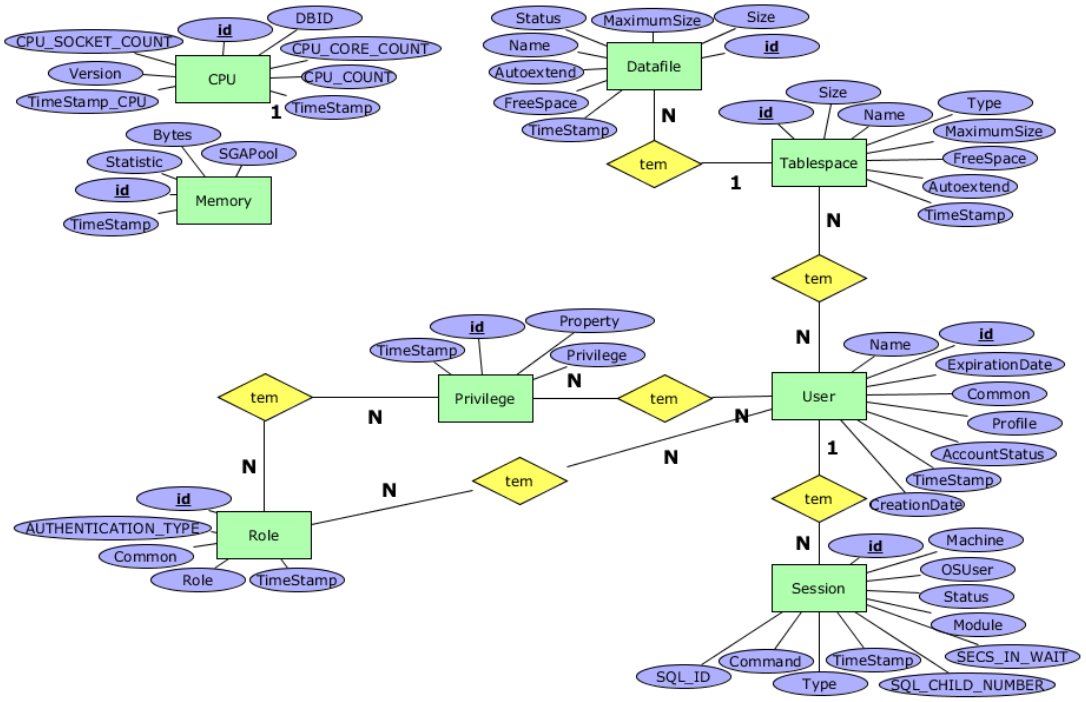

Figura 1 - Modelo conceptual

Analisando todas as vertentes que poderiam ter interesse para a monitorização do estado da Base de Dados, o grupo selecionou as seguintes: *CPU*, *Memory*, *Datafile*, *Tablespace*, *User*, *Session*, *Privilege* e *Role*. Tanto o *CPU* como a *Memory* existem por si só, isto é, não estão relacionados com as outras entidades do modelo. No entanto, quando lidamos com *Tablespaces* temos *Datafiles* relacionados. Assim como temos, também, *Users* que lhes podem aceder. Estes possuem uma *Session* na Base de Dados e têm a si atribuídos *Privileges* e *Roles*.

Os atributos das entidades são dados que foram retirados de várias *views* do sistema da Base de Dados. Os atributos da entidade *CPU* foram, portanto, retirados da *view DBA_CPU_USAGE_STATISTICS*, que indica algumas estatísticas do *CPU*, tais como o número de *CPUs* (*CPU_COUNT*) e o número de *cores* de *CPU* (*CPU_CORE_*COUNT) da Base de Dados. Estes dados informam-nos acerca do estado do uso do *CPU*.

Por outro lado, da *Memory* pretendeu-se extrair informação acerca do espaço lá alocado, a partir da view *V$INMEMORY_AREA*.

Os Tablespaces e os Datafiles contêm informação muito semelhante, baseando-se, pois, em aspetos, tais como o espaço livre (*FreeSpace*), o espaço total (*Size*) e se é *Autoextend* ou não, entre outros. Estes dados foram retirados das *views DBA_TABLESPACES* e *DBA_DATA_FILES* (em conjunção com a *view DBA_FREE_SPACE*), respetivamente. Deste modo, temos acesso às características dos *Tablespaces* e *Datafiles* da Base de Dados em questão.

O *User* tem a si associado dados relativos à sua conta na Base de Dados, como, por exemplo, o seu nome identificador (*Name*), o estado da conta (*AccountStatus*) e data de criação (*CreationDate*), entre outros. Estes dados foram extraídos da *view DBA_USERS* e visam caracterizar cada um dos utilizadores que têm acesso a esta Base de Dados e às suas componentes.

A *Session*, que corresponde a cada sessão dos utilizadores da Base de Dados, possui várias informações acerca das sessões atuais. Podemos verificar qual o comando em progresso (*Command*), bem como o sistema operativo da máquina da sessão que está a aceder à Base de Dados (*Machine*). Este conjunto de dados foi acedido através da *system view V$SESSION* e têm o intuito de nos informar acerca do que cada sessão está a fazer naquele momento. Os dados dos privilégios disponíveis para os utilizadores da Base de Dados, entidade *Privilege*, são retirados da vista *DBA_SYS_PRIVS*. Por fim, as informações contidas nos *Roles* da BD foram extraídas a partir da *view DBA_ROLES.* Estas entidades têm como objetivo mostrar-nos quais as permissões de acesso dos utilizadores.

3. Modelo Lógico da Base de Dados

O modelo lógico da base de dados apresentado no modelo concetual anterior foi criado no *SQLDeveloper* e apresenta o seguinte aspeto:

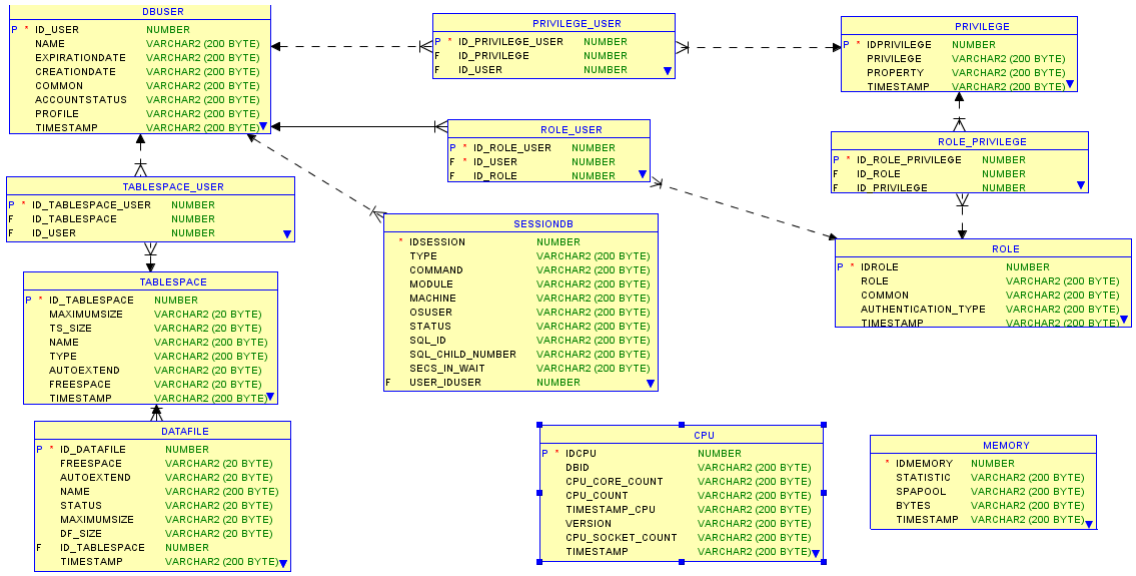

Figura 2 - Modelo lógico da base de dados

3.1. Descrição das entidades

Tabela 1 - Descrição das entidades

Nome da entidade	Descrição	Sinónimos	Contexto
DBUSER	Termo geral que representa todos os *users*	Utilizadores	Cada utilizador é caracterizado pelas suas características, como p.ex., o seu nome e datas de criação e expiração
TABLESPACE	Termo geral que designa todos os	---	Cada *tablespace* é caracterizado, p.ex.,

	tablespace		pelo seu nome, tamanho máximo e tamanho atual
DATAFILE	Termo geral que designa todos os *datafile*	---	Cada *datafile* é caracterizado, p. ex, o seu nome, tamanho máximo e tamanho atual
PRIVILEGE	Termo geral que designa um *privilege*	Privilégio	Cada *privilege* é caracterizado pelo seu nome e propriedade
ROLE	Termo geral que designa um *role*	Papel	Cada *role* é caracterizado pelo seu nome e tipo de autenticação
SESSIONDB	Termo geral que designa uma *session*	Sessão	Cada *session* é caracterizada, p. ex., pelo seu tipo, *status*, máquina.
CPU	Termo geral que representa os dados referentes ao CPU	---	Cada CPU é caracterizada, p.ex., pela sua versão, o *cpu_count, timestamp_cpu*
MEMORY	Termo geral que representa os dados referentes à memória	Memória	Cada memória é caracterizada, p.ex., pelo *sgapool, bytes.*

3.2. Descrição dos relacionamentos

Tabela 2 - Descrição dos relacionamentos

Entidade	Multiplicidade	Relação	Multiplicidade	Entidade
DBUSER	N	Tem	N	TABLESPACE
DBUSER	N	Tem	N	PRIVILEGE
DBUSER	N	Tem	N	ROLE

DBUSER	1	Tem	N	SESSION
TABLESPACE	1	Tem	N	DATAFILE
ROLE	N	Tem	N	PRIVILEGE

3.3. Descrição da entidade DBUSER

Tabela 3 - Descrição dos atributos da DBUser

Entidade	**Atributos**	**Descrição**	**Tipo de dados/Domínio**	**Nulo/Composto**
DBUSER	ID_USER	Identifica de forma única o utilizador	NUMBER	NÃO/NÃO
	NAME	Nome do utilizador	VARCHAR2(200)	SIM/NÃO
	EXPIRATIONDATE	Data de expiração da conta	VARCHAR2(200)	SIM/NÃO
	CREATIONDATE	Data de criação do utilizador	VARCHAR2(200)	SIM/NÃO
	COMMON	Indica se o utilizador é comum	VARCHAR2(200)	SIM/NÃO
	ACCOUNTSTATUS	*Status* da conta do utilizador	VARCHAR2(200)	SIM/NÃO
	PROFILE	Nome de perfil dos recursos do utilizador	VARCHAR2(200)	SIM/NÃO
	TIMESTAMP	Apresenta o *timestamp* da inserção do resgisto	VARCHAR2(200)	SIM/NÃO

3.4. Descrição da entidade TABLESPACE

Tabela 4 - Descrição dos atributos da Tablespace

Entidade	Atributos	Descrição	Tipo de dados/Domínio	Nulo/Composto
TABLESPACE	ID_TABLESPACE	Identifica de forma única o *tablespace*	NUMBER	NÃO/NÃO
	MAXIMUMSIZE	Tamanho máximo do *tablespace*	VARCHAR2(200)	SIM/NÃO
	TS_SIZE	Tamanho do *tablespave*	VARCHAR2(200)	SIM/NÃO
	NAME	Nome do *tablespace*	VARCHAR2(200)	SIM/NÃO
	TYPE	Conteúdo do *tablespace*	VARCHAR2(200)	SIM/NÃO
	AUTOEXTEND	Valor *default* da incrementação do tamanho extensível	VARCHAR2(200)	SIM/NÃO
	FREESPACE	Tamanho livre do *tablespace*	VARCHAR2(200)	SIM/NÃO
	TIMESTAMP	Apresenta o *timestamp* da inserção do resgisto	VARCHAR2(200)	SIM/NÃO

3.5. Descrição da entidade DATAFILE

Tabela 5 - Descrição dos atributos da Datafile

Entidade	Atributos	Descrição	Tipo de dados/Domínio	Nulo/Composto
	ID_DATAFILE	Identifica de forma única o *datafile*	NUMBER	NÃO/NÃO

	FREESPACE	Espaço livre do *datafile*	VARCHAR2(200)	SIM/NÃO
	AUTOEXTEND	Valor *default* da incrementação do tamanho extensível	VARCHAR2(200)	SIM/NÃO
DATAFILE	NAME	Nome do *datafile*	VARCHAR2(200)	SIM/NÃO
	STATUS	*Status* do datafile	VARCHAR2(200)	SIM/NÃO
	MAXIMUMSIZE	Tamanho máximo do *datafile*	VARCHAR2(200)	SIM/NÃO
	DF_SIZE	Tamanho do *datafile*	VARCHAR2(200)	SIM/NÃO
	TIMESTAMP	Apresenta o *timestamp* da inserção do resgisto	VARCHAR2(200)	SIM/NÃO

3.6. Descrição da entidade SESSIONDB

Tabela 6 - Descrição dos atributos da SESSIONDB

Entidade	**Atributos**	**Descrição**	**Tipo de dados/Domínio**	**Nulo/Composto**
	IDSESSION	Identifica de forma única a *session*	NUMBER	NÃO/NÃO
	TYPE	Tipo da *session*	VARCHAR2(200)	SIM/NÃO
	COMMAND	Comando a ser executado	VARCHAR2(200)	SIM/NÃO
SESSIONDB	MODULE	Nome do módulo a ser executado	VARCHAR2(200)	SIM/NÃO
	MACHINE	Nome da	VARCHAR2(200)	SIM/NÃO

		máquina do sistema operativo		
	OSUSER	Nome do cliente do sistema operativo	VARCHAR2(200)	SIM/NÃO
	STATUS	*Status* da *session*	VARCHAR2(200)	SIM/NÃO
	SQL_ID	Identificador da *query* SQL a ser executada	VARCHAR2(200)	SIM/NÃO
	SQL_CHILD_NUMBER	*Child number* da *query* SQL a ser executada	VARCHAR2(200)	SIM/NÃO
	SECS_IN_WAIT	Número de segundos desde o último *wait*	VARCHAR2(200)	SIM/NÃO

3.7. Descrição da entidade PRIVILEGE

Tabela 7 - Descrição dos atributos da Privilege

Entidade	Atributos	Descrição	Tipo de dados/Domínio	Nulo/Composto
PRIVILEGE	IDPRIVILEGE	Identifica de forma única o *privilege*	NUMBER	NÃO/NÃO
	PRIVILEGE	Nome do *privilege* do sistema	VARCHAR2(200)	SIM/NÃO
	PROPERTY	Indica se o *grant* apresenta opção de administrador	VARCHAR2(200)	SIM/NÃO

	TIMESTAMP	Apresenta o *timestamp* da inserção do resgisto	VARCHAR2(200)	SIM/NÃO

3.8. Descrição da entidade ROLE

Tabela 8 - Descrição dos atributos da Role

Entidade	Atributos	Descrição	Tipo de dados/Domínio	Nulo/Composto
ROLE	IDROLE	Identifica de forma única o *role*	NUMBER	NÃO/NÃO
	ROLE	Nome do *role*	VARCHAR2(200)	SIM/NÃO
	COMMON	Indica se o *role* é comum	VARCHAR2(200)	SIM/NÃO
	AUTHENTICATION_TYPE	Tipo de autenticação	VARCHAR2(200)	SIM/NÃO
	TIMESTAMP	Apresenta o *timestamp* da inserção do resgisto	VARCHAR2(200)	SIM/NÃO

3.9. Descrição da entidade CPU

Tabela 9 - Descrição dos atributos do CPU

Entidade	Atributos	Descrição	Tipo de dados/Domínio	Nulo/Composto
	IDCPU	Identifica de forma única o CPU	NUMBER	NÃO/NÃO
	DBID	Identifica de forma única	VARCHAR2(200)	SIM/NÃO

		a BD		
	CPU_CORE_COUNT	Contagem dos *cores* do CPU da BD	VARCHAR2(200)	SIM/NÃO
CPU	CPU_COUNT	Contagem do CPU da BD	VARCHAR2(200)	SIM/NÃO
	TIMESTAMP_CPU	*Timestamp* da mudança do valor de CPU	VARCHAR2(200)	SIM/NÃO
	VERSION	Versão da BD	VARCHAR2(200)	SIM/NÃO
	CPU_SOCKET_COUNT	Contagem dos *sockets* do CPU	VARCHAR2(200)	SIM/NÃO
	TIMESTAMP	Apresenta o *timestamp* da inserção do resgisto	VARCHAR2(200)	SIM/NÃO

3.10. Descrição da entidade MEMORY

Tabela 10 - Descrição dos atributos da Memory

Entidade	**Atributos**	**Descrição**	**Tipo de dados/Domínio**	**Nulo/Composto**
MEMORY	IDMEMORY	Identifica de forma única a *memory*	NUMBER	NÃO/NÃO
	STATISTIC	Total de memória alocada na *pool*	VARCHAR2(200)	SIM/NÃO
	SPAPOOL	Nome das *pools*	VARCHAR2(200)	SIM/NÃO
	BYTES	Total de memória a ser usada na	VARCHAR2(200)	SIM/NÃO

		pool		
	TIMESTAMP	Apresenta o *timestamp* da inserção do registo	VARCHAR2(200)	SIM/NÃO

4. Implementação da Base de Dados

De modo a realizar este trabalho prático, a primeira ação a ser executada foi a criação de *tablespaces* e *datafiles*, de modo a que seja possível criar a base de dados desejada. Assim, estando conectados como administrador de base de dados ("*sys as dba*") críamos dois *tablespaces*, sendo um deles temporário, nos quais, será criada a base de dados. De seguida, apresenta-se o *script* de criação dos dois *tablespaces:*

```
create tablespace aebd_project
datafile '\u01\app\oracle\oradata\orcl12\orcl\aebd_project_01.dbf'
size 100M;

create temporary tablespace aebd_project_temp
tempfile '\u01\app\oracle\oradata\orcl12\orcl\aebd_project_temp.dbf'
size 50M
autoextend on;
```

Figura 3 - Criação dos tablespaces

De seguida, fez-se a criação dos utilizadores que irão trabalhar na base de dados, sendo eles os quatro intervenientes do grupo e associá-los aos *tablespaces aebd_project* e *aebd_project_temp,* como se pode verificar na seguinte figura:

```
create user Bruno
identified by 1234
default tablespace aebd_project
temporary tablespace aebd_project_temp
quota 10M on aebd_project
account unlock;

create user Luis
identified by 1234
default tablespace aebd_project
temporary tablespace aebd_project_temp
quota 10M on aebd_project
account unlock;

create user Maria
identified by 1234
default tablespace aebd_project
temporary tablespace aebd_project_temp
quota 10M on aebd_project
account unlock;

create user Joao
identified by 1234
default tablespace aebd_project
temporary tablespace aebd_project_temp
quota 10M on aebd_project
account unlock;
```

Figura 4 - Criação dos users

O terceiro passo passou por criar uma conexão para ser usada pelos utilizadores que pretendam aceder à base de dados, como se pode ver na seguinte figura:

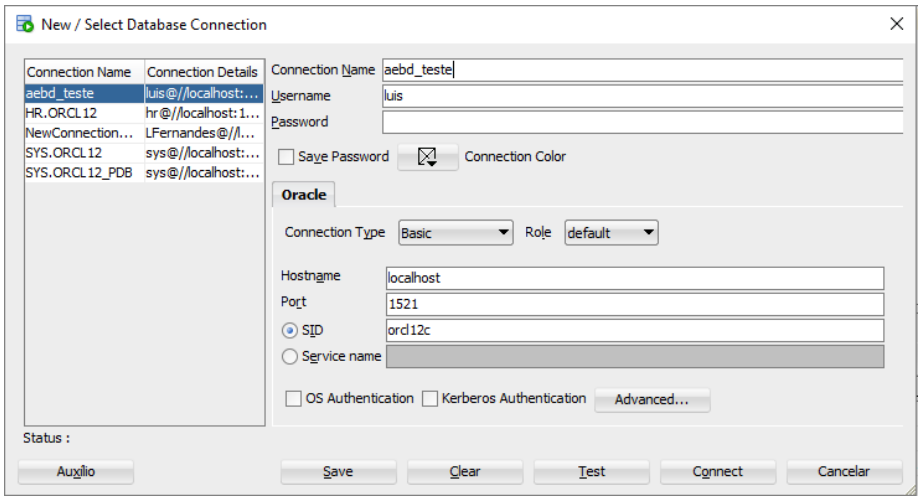

Figura 5 - Criação da conexão

Finalmente, o administrador de base de dados garante permissões aos diversos utilizadores, de modo a que eles possam criar objetos Oracle. Na seguinte figura apresenta-se um exemplo em que se atribuem permissões ao utilizador “Luis”:

```
grant connect to Luis;
grant create table, create view, create trigger to Luis;
grant drop any table to luis;
grant create sequence to luis;
```

Figura 6 - Atribuição de permissões

Neste momento, os utilizadores já podem criar objetos Oracle (tabelas, *triggers*, *views*, *sequences),* estando os *scripts* de criação destes objetos inseridos no fim do ficheiro, nos anexos.

5. Preenchimento da Base de Dados com recurso a programa em Java

5.1. Conexões

Como primeiro passo na implementação deste programa em Java, foi necessário que se estabelecessem conexões a ambas as bases de dados, por um lado estabelecer conexão com a base de dados onde querermos recolher informações, por outro conectar à base de dados onde vamos guardar as mesmas.

Para este efeito foram utilizadas duas classes *OracleConnection.java* e *StatusConnection.java*, que irão, respetivamente, estabelecer as ligações em cima referidas.

Na primeira são utilizados os seguintes argumentos:

```
public static final String DB_DRIVER = "oracle.jdbc.driver.OracleDriver";
public static final String DB_CONNECTION = "jdbc:oracle:thin:@localhost:1521:orcl12c";
public static final String DB_USER = "sys as sysdba";
public static final String DB_PASSWORD = "oracle";
```

Figura 7 - Argumentos da *OracleConnection.java*

Para a segunda é utilizado um dos utilizadores criados anteriormente, respetiva password, mantendo-se os restantes argumentos:

```
public static final String DB_DRIVER = "oracle.jdbc.driver.OracleDriver";
public static final String DB_CONNECTION = "jdbc:oracle:thin:@localhost:1521:orcl12c";
public static final String DB_USER = "luis";
public static final String DB_PASSWORD = "1234";
```

Figura 8 - Argumentos da *StatusConnection.java*

5.2. Gets

Posteriormente à criação da base de dados relacional que irá albergar os dados recolhidos, foi necessário preenche-la, e, para isso, foi criado um programa em Java que irá cumprir tal efeito.

De seguida vão ser apresentados alguns exemplos dos métodos mais importantes do programa que serão suficientes para explicar a lógica do mesmo.

```
public static ResultSet getPrivilege() {

    ResultSet rs = null;

    try {
        PreparedStatement ps = c.prepareStatement("SELECT GRANTEE, PRIVILEGE, ADMIN_OPTION FROM DBA_SYS_PRIVS");
        rs = ps.executeQuery();
    } catch (SQLException ex) {
        ex.printStackTrace();
    }

    return rs;
}
```

Figura 9 - Exemplo método *getPrivilege()*.

Neste método podemos vislumbrar um exemplo de como são efetuadas as recolhas de informação através das *views* de administração, neste caso utilizando a *DBA_SYS_PRIVS* para recolher privilégios associados a um *user* (valores da coluna *GRANTEE*), o que nos permitirá preencher também a tabela *PRIVILEGE_USER*, correspondente à relação N:N estabelecida no modelo conceptual entre as entidades *Privilege* e *User*, sem recorrer a *views* diferentes.

Existem, no entanto, outros casos em que tal não é possível, pelo que, pegando no exemplo do caso representado em seguida, apesar da recolha de informação correspondente aos *roles* e aos *users* nas *views DBA_ROLES* e *DBA_USERS*, respetivamente, foi necessário o recurso a uma terceira *view* (*DBA_ROLE_PRIVS*) para preencher a tabela *ROLE_USER*, tabela correspondente à relação N:N estabelecida no modelo conceptual entre as entidades *Role* e *User*.

```
public static ResultSet getRoleUser() {

    ResultSet rs = null;

    try {
        PreparedStatement ps = c.prepareStatement("SELECT GRANTEE, GRANTED_ROLE FROM DBA_ROLE_PRIVS");
        rs = ps.executeQuery();
    } catch (SQLException ex) {
        ex.printStackTrace();
    }

    return rs;
}
```

Figura 10 - Exemplo método *getRoleUser()*.

5.3. Sets

Posto isto, foi necessário, de seguida, que esta informação fosse inserida na base de dados criada para esse efeito, havendo assim a necessidade de um outro tipo de métodos, no projeto pertencentes à classe *Sets.java* e que terão de seguida o seu funcionamento explicitado em pseudocódigo:

```
public static void setPrivilege(ResultSet resultados) {
            while (resultados.next()) {
                Privilege p = criaResultado();
                List<Privilege> privs = insereResultado(p);
            }
            for(Privilege p : privs) {
                executaQuery("insert into privilege (privilege, property, timestamp) values (p.getPrivilege(), p.getProperty(), timestamp)");
                int idUser = executaQuery("SELECT ID_USER FROM DBUSER WHERE NAME = user AND TIMESTAMP = timestamp");
                int idPriv = executaQuery("SELECT IDPRIVILEGE FROM PRIVILEGE WHERE PRIVILEGE = privilege AND PROPERTY = property AND TIMESTAMP = timestamp");
                executaQuery("INSERT INTO PRIVILEGE_USER VALUES (null, idUser, idPriv)");
            }
}
```

Figura 11 - Exemplo de pseudocódigo para método *setPrivilege()*.

Mais uma vez, tal como explicado anteriormente, a *view DBA_SYS_PRIVS* continha informação necessária para que fosse possível preencher as tabelas *PRIVILEGE* e *PRIVILEGE_USER* sem recorrer a uma outra *view*, algo que nem sempre foi possível.

Para os casos mencionados em que não foi possível efetuar o preenchimento das tabelas e respetivas tabelas de relacionamento, a operação consistiu no preenchimento das tabelas referentes às entidades e posteriormente, de forma análoga à anteriormente explicada, o preenchimento das tabelas referentes aos relacionamentos N:N (foi, por exemplo, efetuado o preenchimento das tabelas *ROLE* e *DBUSER* e posteriormente, com recurso à *view* que se apresenta na Fig.2, foi preenchida a tabela referente ao relacionamento *ROLE_USER*).

6. API REST

Tal como para o programa em Java, foi necessário inicialmente estabelecer uma conexão, utilizando para isso uma classe *Connect.java*, esta conexão servirá para fazer os pedidos à base de dados (comandos SQL).

Estabelecida a conexão, bastou um conjunto de métodos para que o JSON fosse criado, com recurso a *um server Apache Tomcat* para o disponibilizar localmente no endereço *localhost:8080/WebApp/webresources/generic/*XXX, onde XXX representa a tabela que se pretende ver, por exemplo *role*.

Em baixo é mostrado um dos métodos utilizados, sendo que os restantes são análogos ao mencionado.

```
@GET
@Produces(javax.ws.rs.core.MediaType.APPLICATION_JSON)
@Path("role")
public String getRoleJson() throws SQLException {

    PreparedStatement stmt = c.prepareStatement("SELECT * FROM Role");
    ResultSet rs = stmt.executeQuery();
    StringBuilder sb = new StringBuilder();

    try {

        sb.append("[");
        while(rs.next()) {

            Role roleObj = new Role();
            roleObj.setId(rs.getInt("idrole"));
            roleObj.setRole(rs.getString("role"));
            roleObj.setCommon(rs.getString("common"));
            roleObj.setAuthentication_type(rs.getString("authentication_type"));
            roleObj.setTimeStamp(rs.getString("timestamp"));
            sb.append(roleObj.toString());
        }
        sb.setCharAt(sb.length()-1, ']');

    } catch (SQLException e) {
        e.printStackTrace();
    }

    return sb.toString();
}
```

Figura 12 - Método getRoleJson()

Este método guarda o que recebe do pedido à base de dados num objeto do tipo que foi buscar, no caso apresentado um objeto do tipo *Role*, e posteriormente dá uso ao método *toString()* da classe *Role.java*, neste caso mais uma vez, para formar o JSON a apresentar.

```
@Override
    public String toString() {
        return "{\"id\":\"" + id + "\", \"role\":\"" + role + "\", \"common\":\"" + common
            + "\", \"authentication_type\":\"" + authentication_type + "\", \"timeStamp\":\"" + timeStamp + "\"},";
    }
```

Figura 13 - Método toString() da classe Role.java

O ficheiro é gerado e apresentado da seguinte forma:

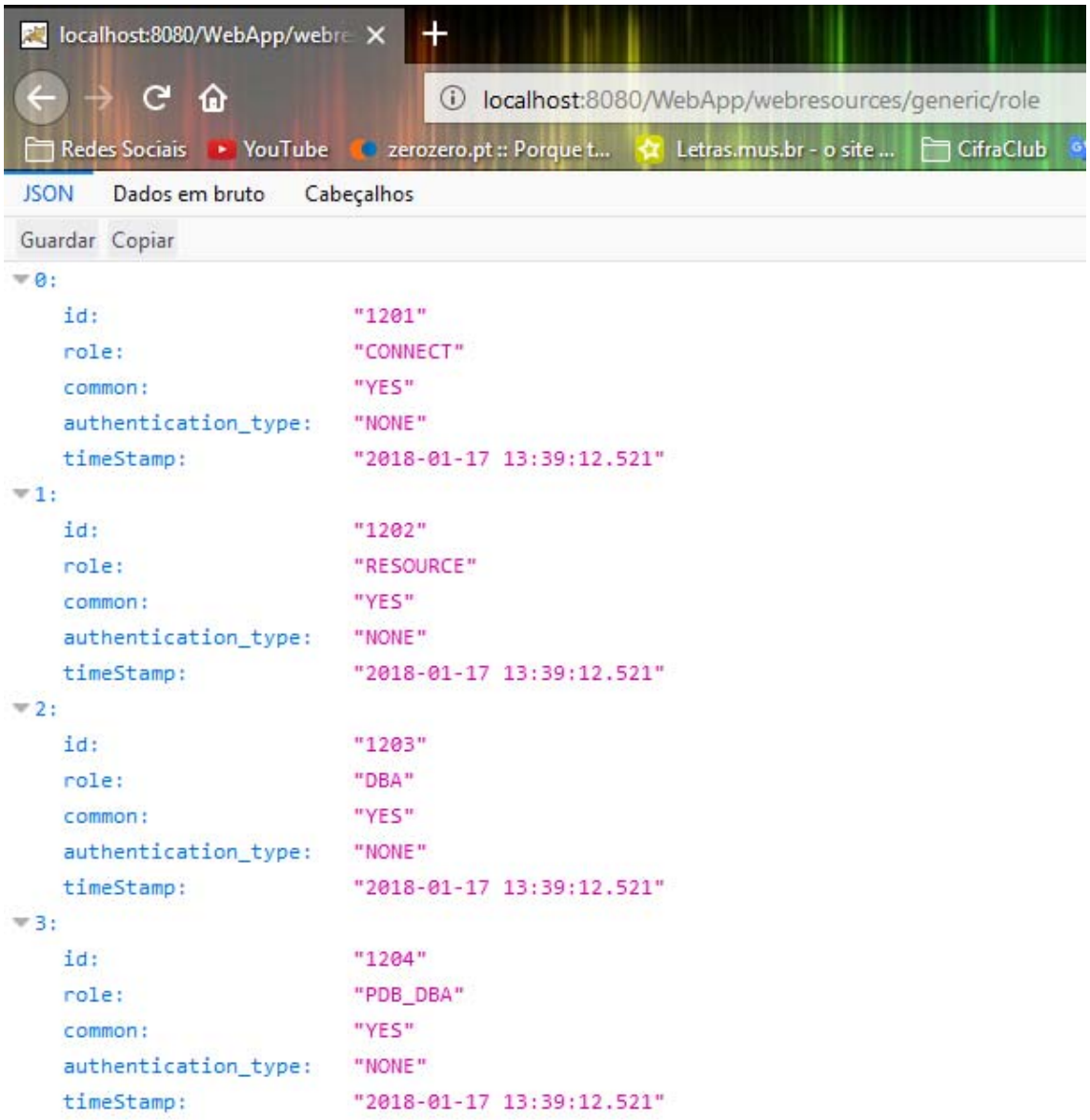

Figura 14 - Exemplo de JSON gerado

7. HTML

Antes da conclusão final do trabalho prático, foi-nos pedido que criássemos uma interface WEB, desenvolvida em HTML5, de modo a sermos capazes de apresentar os elementos presentes na nossa base de dados, de uma forma mais apelativa aos utilizadores interessados.

Decidimos que a interface deveria incluir uma página principal onde dávamos a conhecer o propósito para a criação da mesma, e onde incluiríamos uma hiperligação para os conteúdos consultáveis da nossa base de dados, nomeadamente os registos das tabelas *datafiles, session, memory, CPU, user, tablespace, privilege* e *role*, assim como as relações entre tabelas, que por sua vez geram em termos de *schema*, novas tabelas, como por exemplo, as sessões por utilizador, como se pode ver na **Figura 15**. Por sua vez, cada uma destas páginas iria conter em formato tabular os ditos conteúdos consultáveis da nossa base de dados, como se pode ver na **Figura 16**. Esses conteúdos, retirados através da API REST seriam submetidos a um *parsing* pois o seu formato, JSON, não corresponde aquele que pretendemos apresentar como resultado final.

- Homepage;
    - Apresentação do grupo e descrição do trabalho;
    - Conteúdo (Hiperligação);
        - → Datafile (Hiperligação);
            - Tabela de entradas;
        - → Session (Hiperligação);
            - Tabela de entradas;
        - → Memory (Hiperligação);
            - Tabela de entradas;
        - → CPU (Hiperligação);
            - Tabela de entradas;
        - → User (Hiperligação);
            - Tabela de entradas;
        - → Tablespace (Hiperligação);
            - Tabela de entradas;
        - → Privilege (Hiperligação);
            - Tabela de entradas;
        - → Role (Hiperligação);
            - Tabela de entradas;
        - → …

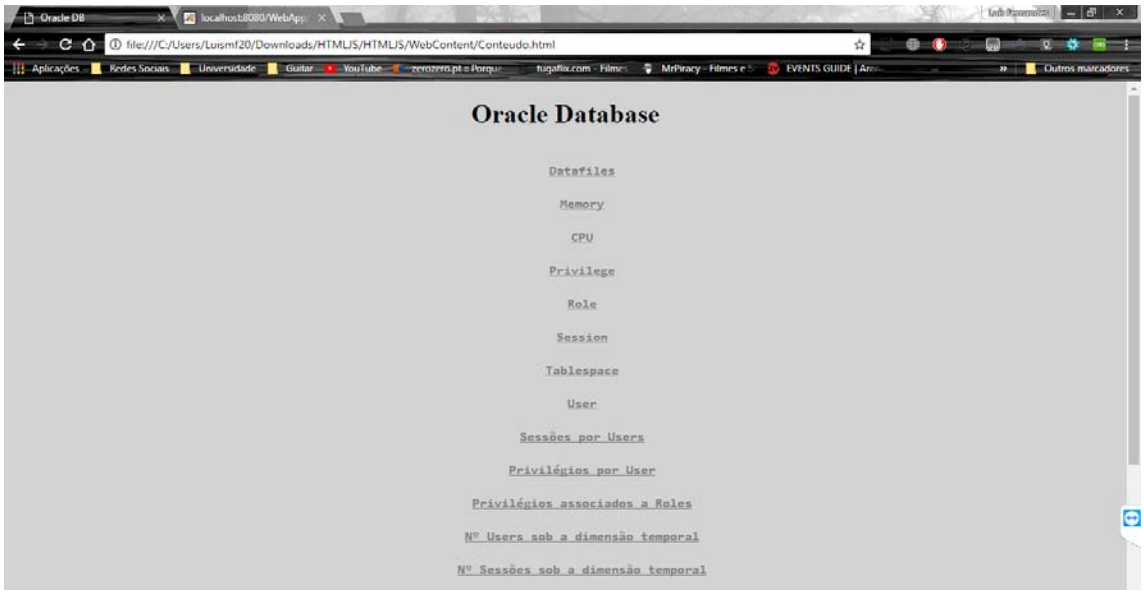

Figura 15 - Página Conteúdos

id	role	common	authentication_type	timeStamp
2548	DV_REALM_RESOURCE	YES	NONE	2018-01-18 17:00:07.638
2547	DV_POLICY_OWNER	YES	NONE	2018-01-18 17:00:07.638
2546	DV_DATAPUMP_NETWORK_LINK	YES	NONE	2018-01-18 17:00:07.638
2545	DV_AUDIT_CLEANUP	YES	NONE	2018-01-18 17:00:07.638
2544	DV_GOLDENGATE_REDO_ACCESS	YES	NONE	2018-01-18 17:00:07.638
2543	DV_XSTREAM_ADMIN	YES	NONE	2018-01-18 17:00:07.638
2542	DV_GOLDENGATE_ADMIN	YES	NONE	2018-01-18 17:00:07.638
2541	DV_STREAMS_ADMIN	YES	NONE	2018-01-18 17:00:07.638
2540	DV_PATCH_ADMIN	YES	NONE	2018-01-18 17:00:07.638
2539	DV_PUBLIC	YES	NONE	2018-01-18 17:00:07.638
2538	DV_ACCTMGR	YES	NONE	2018-01-18 17:00:07.638
2537	DV_OWNER	YES	NONE	2018-01-18 17:00:07.638
2536	DV_ADMIN	YES	NONE	2018-01-18 17:00:07.638
2535	DV_MONITOR	YES	NONE	2018-01-18 17:00:07.638
2534	DV_REALM_OWNER	YES	NONE	2018-01-18 17:00:07.638
2533	LBAC_DBA	YES	NONE	2018-01-18 17:00:07.638

Figura 16 - Página Roles

Escolhemos a linguagem de programação JavaScript, para procedermos ao desenvolvimento de tal interface e transformação dos elementos do formato JSON, para o formato tabular. Esta escolha deveu-se à facilidade da inclusão de elementos desta linguagem no desenvolvimento WEB. As páginas em concreto, foram desenvolvidas seguindo a

linguagem HTML. Recorremos a duas funções escritas em JavaScript para fazer a conversão dos JSON, para tabelas. Estas funções incluem na sua implementação as *tags* HTML, para a inclusão das tabelas na página, a colocação de cada um dos valores lidos do ficheiro JSON na respetiva célula da tabela correspondente, assim como um cálculo do total de colunas que a mesma irá ter de apresentar. Este número corresponde ao número de colunas (atributos) de cada uma das tabelas da nossa base de dados.

A leitura do ficheiro JSON é feita através da opção disponível para o efeito incluída na biblioteca de JavaScript, jQuery. O jQuery permite a inclusão de escrita de funções e manipulação de documentos, dentro da própria implementação de criação da página HTML, do lado do cliente. Deste modo, apenas tivemos que usar esta biblioteca para ler o JSON diretamente de um URL, “gerado” pela API REST (depois de concedermos permissões para tal operação no lado do servidor), e de seguida, usar estes resultados como parâmetro passado às funções desenvolvidas em JavaScript acima descritas.

Após este processo estar concluído, os objetivos traçados estão devidamente cumpridos, visto que somos capazes de consultar de uma maneira muito mais amigável e simplificada os conteúdos da nossa base de dados, quando comparada com uma tentativa de consulta do ficheiro JSON inalterado.

8. Conclusões

O grupo olhou para a Base de Dados segundo a perspetiva de um administrador, selecionando, então, todos os seus parâmetros mais relevantes indicativos de *performance*, desde o uso do CPU, passando pela memória, *tablespaces*, *datafiles*, até às informações sobre os utilizadores, as suas sessões, *roles* e privilégios associados.

Estes dados são recolhidos intermitentemente, logo podem ser analisados por um ponto de vista temporal. Assim, pode-se observar o desempenho da Base de Dados ao longo do tempo de execução do programa e tecer as devidas conclusões. Este processo é útil para um administrador de Base de Dados, pois é o responsável por realizar uma monitorização atenta da BD, visando o seu bom desempenho e funcionamento.

Em suma, foi adquirida uma melhor compreensão acerca dos vários dados das *views* da Base de Dados que contribuem para uma exploração e análise do desempenho da BD.

Anexos

Em seguida, apresentar-se-ão alguns anexos relevantes do desenvolvimento do projeto, nomeadamente o *script* de criação de tabelas na Base de Dados, bem como a criação de *triggers*, índices, entre outros, que devido à sua dimensão estão anexados ao relatório do projeto.

A. Criação das Tabelas e dos Triggers

```
--------------------------------------------------------
--  DDL for Table CPU
--------------------------------------------------------

  CREATE TABLE "LUIS"."CPU"
   (   "IDCPU" NUMBER,
       "DBID" VARCHAR2(200 BYTE),
       "CPU_CORE_COUNT" VARCHAR2(200 BYTE),
       "CPU_COUNT" VARCHAR2(200 BYTE),
       "TIMESTAMP_CPU" VARCHAR2(200 BYTE),
       "VERSION" VARCHAR2(200 BYTE),
       "CPU_SOCKET_COUNT" VARCHAR2(200 BYTE),
       "TIMESTAMP" VARCHAR2(200 BYTE)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table MEMORY
--------------------------------------------------------

  CREATE TABLE "LUIS"."MEMORY"
   (  "IDMEMORY" NUMBER,
  "STATISTIC" VARCHAR2(200 BYTE),
  "SPAPOOL" VARCHAR2(200 BYTE),
  "BYTES" VARCHAR2(200 BYTE),
  "TIMESTAMP" VARCHAR2(200 BYTE)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table TABLESPACE
--------------------------------------------------------

  CREATE TABLE "LUIS"."TABLESPACE"
   (  "ID_TABLESPACE" NUMBER,
  "MAXIMUMSIZE" VARCHAR2(20 BYTE),
  "TS_SIZE" VARCHAR2(20 BYTE),
  "NAME" VARCHAR2(200 BYTE),
```

```
  "TYPE" VARCHAR2(200 BYTE),
  "AUTOEXTEND" VARCHAR2(20 BYTE),
  "FREESPACE" VARCHAR2(20 BYTE),
  "TIMESTAMP" VARCHAR2(200 BYTE)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table DATAFILE
--------------------------------------------------------

  CREATE TABLE "LUIS"."DATAFILE"
   (   "ID_DATAFILE" NUMBER,
       "FREESPACE" VARCHAR2(20 BYTE),
       "AUTOEXTEND" VARCHAR2(20 BYTE),
       "NAME" VARCHAR2(200 BYTE),
       "STATUS" VARCHAR2(20 BYTE),
       "MAXIMUMSIZE" VARCHAR2(20 BYTE),
       "DF_SIZE" VARCHAR2(20 BYTE),
       "ID_TABLESPACE" NUMBER,
       "TIMESTAMP" VARCHAR2(200 BYTE)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table DBUSER
--------------------------------------------------------

  CREATE TABLE "LUIS"."DBUSER"
   (   "ID_USER" NUMBER,
       "NAME" VARCHAR2(200 BYTE),
       "EXPIRATIONDATE" VARCHAR2(200 BYTE),
       "CREATIONDATE" VARCHAR2(200 BYTE),
       "COMMON" VARCHAR2(200 BYTE),
       "ACCOUNTSTATUS" VARCHAR2(200 BYTE),
       "PROFILE" VARCHAR2(200 BYTE),
       "TIMESTAMP" VARCHAR2(200 BYTE)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table SESSIONDB
--------------------------------------------------------

  CREATE TABLE "LUIS"."SESSIONDB"
   (  "IDSESSION" NUMBER,
  "TYPE" VARCHAR2(200 BYTE),
```

```
  "COMMAND" VARCHAR2(200 BYTE),
  "MODULE" VARCHAR2(200 BYTE),
  "MACHINE" VARCHAR2(200 BYTE),
  "OSUSER" VARCHAR2(200 BYTE),
  "STATUS" VARCHAR2(200 BYTE),
  "SQL_ID" VARCHAR2(200 BYTE),
  "SQL_CHILD_NUMBER" VARCHAR2(200 BYTE),
  "SECS_IN_WAIT" VARCHAR2(200 BYTE),
  "USER_IDUSER" NUMBER,
  "TIMESTAMP" VARCHAR2(200 BYTE)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table PRIVILEGE
--------------------------------------------------------

  CREATE TABLE "LUIS"."PRIVILEGE"
   (   "IDPRIVILEGE" NUMBER,
       "PRIVILEGE" VARCHAR2(200 BYTE),
       "PROPERTY" VARCHAR2(200 BYTE),
       "TIMESTAMP" VARCHAR2(200 BYTE)
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table PRIVILEGE_USER
--------------------------------------------------------

  CREATE TABLE "LUIS"."PRIVILEGE_USER"
   (   "ID_PRIVILEGE_USER" NUMBER,
       "ID_PRIVILEGE" NUMBER,
       "ID_USER" NUMBER
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table ROLE
--------------------------------------------------------

  CREATE TABLE "LUIS"."ROLE"
   (   "IDROLE" NUMBER,
       "ROLE" VARCHAR2(200 BYTE),
       "COMMON" VARCHAR2(200 BYTE),
       "AUTHENTICATION_TYPE" VARCHAR2(200 BYTE),
       "TIMESTAMP" VARCHAR2(200 BYTE)
   ) SEGMENT CREATION IMMEDIATE
```

```
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table ROLE_PRIVILEGE
--------------------------------------------------------

  CREATE TABLE "LUIS"."ROLE_PRIVILEGE"
   (   "ID_ROLE_PRIVILEGE" NUMBER,
       "ID_ROLE" NUMBER,
       "ID_PRIVILEGE" NUMBER
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table ROLE_USER
--------------------------------------------------------

  CREATE TABLE "LUIS"."ROLE_USER"
   (   "ID_ROLE_USER" NUMBER,
       "ID_USER" NUMBER,
       "ID_ROLE" NUMBER
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Table TABLESPACE_USER
--------------------------------------------------------

  CREATE TABLE "LUIS"."TABLESPACE_USER"
   (   "ID_TABLESPACE_USER" NUMBER,
       "ID_TABLESPACE" NUMBER,
       "ID_USER" NUMBER
   ) SEGMENT CREATION IMMEDIATE
  PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index CPU_PK
--------------------------------------------------------

  CREATE UNIQUE INDEX "LUIS"."CPU_PK" ON "LUIS"."CPU" ("IDCPU")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
```

```
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index DATAFILE_PK
--------------------------------------------------------

  CREATE    UNIQUE    INDEX    "LUIS"."DATAFILE_PK"    ON    "LUIS"."DATAFILE"
("ID_DATAFILE")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index ID_USER
--------------------------------------------------------

  CREATE UNIQUE INDEX "LUIS"."ID_USER" ON "LUIS"."DBUSER" ("ID_USER")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index MEMORY_PK
--------------------------------------------------------

  CREATE UNIQUE INDEX "LUIS"."MEMORY_PK" ON "LUIS"."MEMORY" ("IDMEMORY")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index PRIVILEGE_PK
--------------------------------------------------------

  CREATE    UNIQUE    INDEX    "LUIS"."PRIVILEGE_PK"    ON    "LUIS"."PRIVILEGE"
("IDPRIVILEGE")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index PRIVILEGE_USER_PK
--------------------------------------------------------

  CREATE  UNIQUE  INDEX  "LUIS"."PRIVILEGE_USER_PK"  ON  "LUIS"."PRIVILEGE_USER"
("ID_PRIVILEGE_USER")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index ROLE_PK
--------------------------------------------------------
```

```
  CREATE UNIQUE INDEX "LUIS"."ROLE_PK" ON "LUIS"."ROLE" ("IDROLE")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index ROLE_PRIVILEGE_PK
--------------------------------------------------------

  CREATE UNIQUE INDEX "LUIS"."ROLE_PRIVILEGE_PK" ON "LUIS"."ROLE_PRIVILEGE"
("ID_ROLE_PRIVILEGE")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index ROLE_USER_PK
--------------------------------------------------------

  CREATE UNIQUE INDEX "LUIS"."ROLE_USER_PK" ON "LUIS"."ROLE_USER"
("ID_ROLE_USER")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index SESSION_PK
--------------------------------------------------------

  CREATE UNIQUE INDEX "LUIS"."SESSION_PK" ON "LUIS"."SESSIONDB" ("IDSESSION")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index TABLESPACE_PK
--------------------------------------------------------

  CREATE UNIQUE INDEX "LUIS"."TABLESPACE_PK" ON "LUIS"."TABLESPACE"
("ID_TABLESPACE")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Index TABLESPACE_USER_PK
--------------------------------------------------------

  CREATE UNIQUE INDEX "LUIS"."TABLESPACE_USER_PK" ON "LUIS"."TABLESPACE_USER"
("ID_TABLESPACE_USER")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
```

```
  TABLESPACE "AEBD_PROJECT" ;
--------------------------------------------------------
--  DDL for Trigger CPUID_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."CPUID_TRIG"
before insert on cpu
for each row
     WHEN (new.idcpu is null) begin
    select cpu_seq.nextval
    into :new.idcpu
    from dual;
end;
/
ALTER TRIGGER "LUIS"."CPUID_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger DATAFILE_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."DATAFILE_TRIG"
before insert on datafile
for each row
     WHEN (new.id_datafile is null) begin
    select datafile_seq.nextval
    into :new.id_datafile
    from dual;
end;
/
ALTER TRIGGER "LUIS"."DATAFILE_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger USER_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."USER_TRIG"
before insert on dbuser
for each row
     WHEN (new.id_user is null) begin
    select user_seq.nextval
    into :new.id_user
    from dual;
end;
/
ALTER TRIGGER "LUIS"."USER_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger MEMORY_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."MEMORY_TRIG"
before insert on memory
for each row
     WHEN (new.idmemory is null) begin
    select memory_seq.nextval
    into :new.idmemory
    from dual;
end;
/
ALTER TRIGGER "LUIS"."MEMORY_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger PRIVILEGE_TRIG
```

```
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."PRIVILEGE_TRIG"
before insert on privilege
for each row
     WHEN (new.idprivilege is null) begin
    select privilege_seq.nextval
    into :new.idprivilege
    from dual;
end;
/
ALTER TRIGGER "LUIS"."PRIVILEGE_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger PRIVILEGE_USER_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."PRIVILEGE_USER_TRIG"
before insert on privilege_user
for each row
     WHEN (new.id_privilege_user is null) begin
    select privilege_user_seq.nextval
    into :new.id_privilege_user
    from dual;
end;
/
ALTER TRIGGER "LUIS"."PRIVILEGE_USER_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger ROLE_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."ROLE_TRIG"
before insert on role
for each row
     WHEN (new.idrole is null) begin
    select role_seq.nextval
    into :new.idrole
    from dual;
end;
/
ALTER TRIGGER "LUIS"."ROLE_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger ROLE_PRIVILEGE_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."ROLE_PRIVILEGE_TRIG"
before insert on role_privilege
for each row
     WHEN (new.id_role_privilege is null) begin
    select role_privilege_seq.nextval
    into :new.id_role_privilege
    from dual;
end;
/
ALTER TRIGGER "LUIS"."ROLE_PRIVILEGE_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger ROLE_USER_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."ROLE_USER_TRIG"
```

```
before insert on role_user
for each row
     WHEN (new.id_role_user is null) begin
    select role_user_seq.nextval
    into :new.id_role_user
    from dual;
end;
/
ALTER TRIGGER "LUIS"."ROLE_USER_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger SESSION_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."SESSION_TRIG"
before insert on sessiondb
for each row
     WHEN (new.idsession is null) begin
    select session_seq.nextval
    into :new.idsession
    from dual;
end;
/
ALTER TRIGGER "LUIS"."SESSION_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger TABLESPACE_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."TABLESPACE_TRIG"
before insert on tablespace
for each row
     WHEN (new.id_tablespace is null) begin
    select tablespace_seq.nextval
    into :new.id_tablespace
    from dual;
end;
/
ALTER TRIGGER "LUIS"."TABLESPACE_TRIG" ENABLE;
--------------------------------------------------------
--  DDL for Trigger TABLESPACE_USER_TRIG
--------------------------------------------------------

  CREATE OR REPLACE NONEDITIONABLE TRIGGER "LUIS"."TABLESPACE_USER_TRIG"
before insert on tablespace_user
for each row
     WHEN (new.id_tablespace_user is null) begin
    select tablespace_user_seq.nextval
    into :new.id_tablespace_user
    from dual;
end;
/
ALTER TRIGGER "LUIS"."TABLESPACE_USER_TRIG" ENABLE;
--------------------------------------------------------
--  Constraints for Table CPU
--------------------------------------------------------

  ALTER TABLE "LUIS"."CPU" MODIFY ("IDCPU" NOT NULL ENABLE);
  ALTER TABLE "LUIS"."CPU" ADD CONSTRAINT "CPU_PK" PRIMARY KEY ("IDCPU")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
```

```
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT"  ENABLE;
--------------------------------------------------------
--  Constraints for Table DATAFILE
--------------------------------------------------------

  ALTER TABLE "LUIS"."DATAFILE" MODIFY ("ID_DATAFILE" NOT NULL ENABLE);
  ALTER  TABLE  "LUIS"."DATAFILE"  ADD  CONSTRAINT  "DATAFILE_PK"  PRIMARY  KEY
("ID_DATAFILE")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT"  ENABLE;
--------------------------------------------------------
--  Constraints for Table DBUSER
--------------------------------------------------------

  ALTER TABLE "LUIS"."DBUSER" MODIFY ("ID_USER" NOT NULL ENABLE);
  ALTER   TABLE   "LUIS"."DBUSER"   ADD   CONSTRAINT   "ID_USER"   PRIMARY   KEY
("ID_USER")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT"  ENABLE;
--------------------------------------------------------
--  Constraints for Table MEMORY
--------------------------------------------------------

  ALTER TABLE "LUIS"."MEMORY" MODIFY ("IDMEMORY" NOT NULL ENABLE);
  ALTER   TABLE   "LUIS"."MEMORY"   ADD   CONSTRAINT   "ID_MEMORY"   PRIMARY   KEY
("IDMEMORY")
  USING  INDEX  (CREATE  UNIQUE  INDEX  "LUIS"."MEMORY_PK"  ON  "LUIS"."MEMORY"
("IDMEMORY")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" )  ENABLE;
--------------------------------------------------------
--  Constraints for Table PRIVILEGE
--------------------------------------------------------

  ALTER TABLE "LUIS"."PRIVILEGE" MODIFY ("IDPRIVILEGE" NOT NULL ENABLE);
  ALTER  TABLE  "LUIS"."PRIVILEGE"  ADD  CONSTRAINT  "PRIVILEGE_PK"  PRIMARY  KEY
("IDPRIVILEGE")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT"  ENABLE;
--------------------------------------------------------
--  Constraints for Table PRIVILEGE_USER
--------------------------------------------------------

  ALTER  TABLE  "LUIS"."PRIVILEGE_USER"  MODIFY  ("ID_PRIVILEGE_USER"  NOT  NULL
ENABLE);
```

```
  ALTER TABLE "LUIS"."PRIVILEGE_USER" ADD CONSTRAINT "PRIVILEGE_USER_PK"
PRIMARY KEY ("ID_PRIVILEGE_USER")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT"  ENABLE;
--------------------------------------------------------
--  Constraints for Table ROLE
--------------------------------------------------------

  ALTER TABLE "LUIS"."ROLE" MODIFY ("IDROLE" NOT NULL ENABLE);
  ALTER TABLE "LUIS"."ROLE" ADD CONSTRAINT "ROLE_PK" PRIMARY KEY ("IDROLE")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT"  ENABLE;
--------------------------------------------------------
--  Constraints for Table ROLE_PRIVILEGE
--------------------------------------------------------

  ALTER TABLE "LUIS"."ROLE_PRIVILEGE" MODIFY ("ID_ROLE_PRIVILEGE" NOT NULL
ENABLE);
  ALTER TABLE "LUIS"."ROLE_PRIVILEGE" ADD CONSTRAINT "ROLE_PRIVILEGE_PK"
PRIMARY KEY ("ID_ROLE_PRIVILEGE")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT"  ENABLE;
--------------------------------------------------------
--  Constraints for Table ROLE_USER
--------------------------------------------------------

  ALTER TABLE "LUIS"."ROLE_USER" MODIFY ("ID_ROLE_USER" NOT NULL ENABLE);
  ALTER TABLE "LUIS"."ROLE_USER" MODIFY ("ID_USER" NOT NULL ENABLE);
  ALTER TABLE "LUIS"."ROLE_USER" ADD CONSTRAINT "ROLE_USER_PK" PRIMARY KEY
("ID_ROLE_USER")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT"  ENABLE;
--------------------------------------------------------
--  Constraints for Table SESSIONDB
--------------------------------------------------------

  ALTER TABLE "LUIS"."SESSIONDB" MODIFY ("IDSESSION" NOT NULL ENABLE);
  ALTER TABLE "LUIS"."SESSIONDB" ADD CONSTRAINT "ID_SESSION" PRIMARY KEY
("IDSESSION")
  USING INDEX (CREATE UNIQUE INDEX "LUIS"."SESSION_PK" ON "LUIS"."SESSIONDB"
("IDSESSION")
  PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT" )  ENABLE;
--------------------------------------------------------
```

```
--  Constraints for Table TABLESPACE
------------------------------------------------------

  ALTER TABLE "LUIS"."TABLESPACE" MODIFY ("ID_TABLESPACE" NOT NULL ENABLE);
  ALTER TABLE "LUIS"."TABLESPACE" ADD CONSTRAINT "TABLESPACE_PK" PRIMARY KEY
("ID_TABLESPACE")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT"  ENABLE;
------------------------------------------------------
--  Constraints for Table TABLESPACE_USER
------------------------------------------------------

  ALTER TABLE "LUIS"."TABLESPACE_USER" MODIFY ("ID_TABLESPACE_USER" NOT NULL
ENABLE);
  ALTER TABLE "LUIS"."TABLESPACE_USER" ADD CONSTRAINT "TABLESPACE_USER_PK"
PRIMARY KEY ("ID_TABLESPACE_USER")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255 COMPUTE STATISTICS
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "AEBD_PROJECT"  ENABLE;
```

B. Criação das Sequences

```
--------------------------------------------------------
--  DDL for Sequence CPU_SEQ
--------------------------------------------------------

   CREATE     SEQUENCE     "LUIS"."CPU_SEQ"     MINVALUE     1     MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence DATAFILE_SEQ
--------------------------------------------------------

   CREATE    SEQUENCE    "LUIS"."DATAFILE_SEQ"    MINVALUE    1    MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence MEMORY_SEQ
--------------------------------------------------------

   CREATE     SEQUENCE     "LUIS"."MEMORY_SEQ"     MINVALUE     1     MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence PRIVILEGE_SEQ
--------------------------------------------------------

   CREATE    SEQUENCE    "LUIS"."PRIVILEGE_SEQ"    MINVALUE    1    MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence PRIVILEGE_USER_SEQ
--------------------------------------------------------

   CREATE   SEQUENCE   "LUIS"."PRIVILEGE_USER_SEQ"   MINVALUE   1   MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence ROLE_PRIVILEGE_SEQ
--------------------------------------------------------

   CREATE   SEQUENCE   "LUIS"."ROLE_PRIVILEGE_SEQ"   MINVALUE   1   MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence ROLE_SEQ
--------------------------------------------------------
```

```
   CREATE     SEQUENCE     "LUIS"."ROLE_SEQ"     MINVALUE     1     MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence ROLE_USER_SEQ
--------------------------------------------------------

   CREATE    SEQUENCE    "LUIS"."ROLE_USER_SEQ"    MINVALUE    1    MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence SESSION_SEQ
--------------------------------------------------------

   CREATE     SEQUENCE     "LUIS"."SESSION_SEQ"     MINVALUE     1     MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence TABLESPACE_SEQ
--------------------------------------------------------

   CREATE    SEQUENCE    "LUIS"."TABLESPACE_SEQ"    MINVALUE    1    MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence TABLESPACE_USER_SEQ
--------------------------------------------------------

   CREATE   SEQUENCE   "LUIS"."TABLESPACE_USER_SEQ"   MINVALUE   1   MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
--------------------------------------------------------
--  DDL for Sequence USER_SEQ
--------------------------------------------------------

   CREATE     SEQUENCE     "LUIS"."USER_SEQ"     MINVALUE     1     MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER
NOCYCLE  NOKEEP  NOSCALE  GLOBAL ;
```

C. Atribuição das Chaves Estrangeiras

```
-- DATAFILE --
ALTER TABLE DATAFILE
ADD CONSTRAINT DATAFILE_FK1 FOREIGN KEY
(
  ID_TABLESPACE
)
REFERENCES TABLESPACE
(
  ID_TABLESPACE
)
ENABLE;

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

-- PRIVILEGE_USER --

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

ALTER TABLE PRIVILEGE_USER
ADD CONSTRAINT PRIVILEGE_USER_FK1 FOREIGN KEY
(
  ID_PRIVILEGE
)
REFERENCES PRIVILEGE
(
  IDPRIVILEGE
)
ENABLE;

ALTER TABLE PRIVILEGE_USER
ADD CONSTRAINT PRIVILEGE_USER_FK2 FOREIGN KEY
(
  ID_USER
)
REFERENCES DBUSER
(
  ID_USER
)
ENABLE;

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

-- ROLE_PRIVILEGE --

-----------------------------------------------
```

```
ALTER TABLE ROLE_PRIVILEGE
ADD CONSTRAINT ROLE_PRIVILEGE_FK1 FOREIGN KEY
(
  ID_ROLE
)
REFERENCES ROLE
(
  IDROLE
)
ENABLE;

ALTER TABLE ROLE_PRIVILEGE
ADD CONSTRAINT ROLE_PRIVILEGE_FK2 FOREIGN KEY
(
  ID_PRIVILEGE
)
REFERENCES PRIVILEGE
(
  IDPRIVILEGE
)
ENABLE;

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

-- ROLE_USER --

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

ALTER TABLE ROLE_USER
ADD CONSTRAINT ROLE_USER_FK1 FOREIGN KEY
(
  ID_USER
)
REFERENCES DBUSER
(
  ID_USER
)
ENABLE;

ALTER TABLE ROLE_USER
ADD CONSTRAINT ROLE_USER_FK2 FOREIGN KEY
(
  ID_ROLE
)
REFERENCES ROLE
(
  IDROLE
)
ENABLE;

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

-- SESSION --

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

ALTER TABLE SESSIONDB
ADD CONSTRAINT ID_FK_SESSION FOREIGN KEY
(
```

```
  USER_IDUSER
)
REFERENCES DBUSER
(
  ID_USER
)
ENABLE;

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

-- TABLESPACE_USER --

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

ALTER TABLE TABLESPACE_USER
ADD CONSTRAINT TABLESPACE_USER_FK1 FOREIGN KEY
(
  ID_TABLESPACE
)
REFERENCES TABLESPACE
(
  ID_TABLESPACE
)
ENABLE;

ALTER TABLE TABLESPACE_USER
ADD CONSTRAINT TABLESPACE_USER_FK2 FOREIGN KEY
(
  ID_USER
)
REFERENCES DBUSER
(
  ID_USER
)
ENABLE;
```